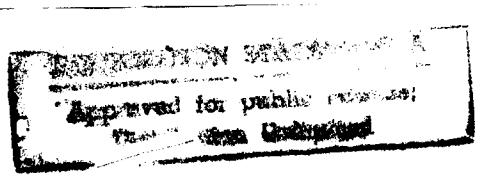
SEISMIC MONITORING WITH SMALL APERTURE ARRAYS UNDER STRONG NOISE CONDITIONS: ALGORITHMS, TECHNIQUE, SYSTEM DESIGN AND EXPERIMENTAL DATA PROCESSING

special project SPC-94-4039

Moscow IRIS Data Analysis Center/

SYNAPSE Science Center

Moscow, May 1995



LIST OF PARTICIPANTS OF THE PROJECT

Alexander Fedorovich Kushnir, Principal investigator
Dr.Sc.,Prof., Senior Researcher of
Moscow IRIS Data Analysis Center
SYNAPSE Science Center &
International Institute for Earthquake Prediction Theory and
Mathematical and Geophysics Russian Academy of Sciences

Leonid Matveevich Haikin
Ph.D., Senior Researcher of Moscow IRIS Data Analysis Centre /
SYNAPSE Science Centre

Lapshin, Victor Mikhailovich,
Ph.D., Chief of laboratory of Institute of Physics of the Earth,
RAS, Moscow

Boris Mihkailovich Shoubik
Ph.D., Senior researcher of Institute of Physics of the Earth,
RAS, Moscow

Evgeny Valerievich Troitsky
Ph.D., Assistant professor of Moscow Technological University of
Communications and Information.

Alexei Borisovich Chulcov
Mst.D., Researcher of Institute of Physics of the Earth,
RAS, Moscow

SYNAPSE address:
SYNAPSE Science Center / Moscow IRIS Data Analysis Center,
Ostozhenka street 13/12 Stroenie 1 a-b, Moscow 121034
Telephone 095-201-2516/
FAX 095-202-6934
E-mail lap@synapse.ru

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1995	Final Report

4. TITLE AND SUBTITLE

Seismic Monitoring with Small Aperture Arrays Under Strong Noise Conditions: Algorithms, Technique, System Design and Experimental Data Processing

5. FUNDING NUMBERS

F6170894W0452

6. AUTHOR(S)

Dr. Alex Kushnir

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Moscow Iris Data Center
Ostozhenka Street
12/13 BLDG I -AB
Moscow 121019
Russia

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
PSC 802 BOX 14
FPO 09499-0200

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

SPC 94-4039

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

A

13. ABSTRACT (Maximum 200 words)

This report results from a contract tasking Moscow Iris Data Center as follows: Investigate array processing techniques and apply them to small seismic events and to events with weak surface wave phases.

14. SUBJECT TERMS

EOARD

15. NUMBER OF PAGES

16. PRICE CODE
N/A

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

CONTENT

PART I. DEVELOPMENT OF STATISTICAL ALGORITHMS AND SOFTWARE FOR SMALL APERTURE ARRAY DATA PROCESSING

PART II. DEVELOPMENT OF PROBLEM ORIENTED PROGRAM PACKAGE FOR SEISMIC ARRAY AND NETWORK DATA ANALYSIS

SUMMARY OF THE PROJECT RESULTS

A subject of the first stage of research was development of advanced techniques for regional seismic monitoring using data from Small Aperture Seismic Arrays (SASA) in areas affected by strong seismic noise. The main idea of the research is the application of an adaptive statistical approach for seismic signal detection and parameter estimation in noisy environments. This approach yields strong SNR enhancement and hence gain in reliability of signal detection and accuracy of parameter estimation in conditions where seismic noise is essentially nonstationary. The implementation of SASA for seismic monitoring allows us to utilize noise coherency features prominent low and intermediate seismic frequencies for noise suppression with the help of the adaptive group filtering (multichannel Wiener filtering) technique. The theoretical background of the study is modern statistical methods for estimating characteristics of multidimensional random time series (e.g., multidimensional ARMA modeling) and recently developed asymptotic techniques for statistically optimal algorithm synthesis and analysis.

A package of adaptive statistical algorithms and programs for processing SASA data was developed and tested on simulated and real data. The package comprises the following processing procedures which are essential in automated monitoring of regional seismicity and underground nuclear tests:

a) adaptive optimal group filtering of array data (with an option for whitening of noise residuals: option distorts a signal waveform but maximally enhances its SNR);

b) multichannel spatial rejection filtering which suppresses interfering waves arriving from assigned directions;

c) adaptive statistically optimal detection of seismic phases;

d) maximum likelihood estimation of onset times of seismic wave phases;

e) adaptive statistically optimal estimation of azimuth and apparent velocity of signal wave obscured by strong coherent noise or interfering waves from another seismic event.

The above algorithms are designed for broad-band array data processing. In this regard they are distinct from the techniques currently used for small aperture seismic array data processing which analyze seismic data in a set of narrow frequency bands. The adaptive statistically optimal algorithms were tested using data recorded at the Scandinavian small aperture

seismic arrays NORSAR, NORESS, ARCESS and FINESA. A special experimental study was performed for an assessment of the effectiveness of adaptive optimal group filtering for enhancing surface waves obscured by long period coherent noise using data from large aperture arrays such as NORSAR and GRAFENBERG. Computer simulations were also performed for testing procedures developed and investigating their potential in comparison with conventional data processing techniques.

The problem oriented programming shell SNDA (Seismic Networks Data Analysis) was developed for the near-real-time automated and interactive processing of data from seismic arrays and networks. It incorporates the above adaptive statistical processing techniques and a set of conventional data analysis procedures. The SNDA can be valuable for different geophysical and ecological monitoring applications connected with the collecting and processing of a large amount of multichannel experimental data.

The SNDA consists of Subsystems for Real Time (RT) and Off-line Interactive multichannel data processing. The RT Subsystem contains the software for accepting a real time multichannel data flow and storing data in a moving time window disc loop. The on-line adaptive beamforming and statistical detection procedures with periodic adaptation for current seismic noise are installed in the RTS. The Interactive SNDA Subsystem provides adaptive statistical processing of seismic multichannel data along with conventional multivariate time series analysis. The data analysis can be performed in the SNDA in multi-user interactive or batch modes. The SNDA includes original graphic software for displaying and interactive treatment of multichannel data, including such capabilities as selection of channels, time zooming, trace scaling, time and amplitude measurement, trace superposition, along with software for 2 and 3 dimensional mapping and X-Y plotting. An original internal command language was developed in the SNDA which enables a user to compose scripts for comprehensive data processing, i.e., for performing a sophisticated sequence of data processing procedures relevant for a given analysis.

SYNAPSE Science Center / Moscow IRIS Data Analysis Center

International Institute for Earthquake Prediction Theory and
Mathematical Geophysics Russian Academy of Sciences

Joint Institute of Physics of the Earth
Russian Academy of Sciences

Part 1

DEVELOPMENT OF STATISTICAL ALGORITHMS AND SOFTWARE FOR SMALL APERTURE ARRAY DATA PROCESSING

Moscow, May 1995

CONTENT

1. Introduction

The advent of a high-resolution and very-broad-band (VBB) digital seismometers [79]have brought great advances to seismology over the past decade. Nowadays the main problem in instrumental seismology is an influence of seismic noise, mainly at low frequencies ($f < 3.$ Hz) of the seismic range. Any observational sites are affected by transient seismic noise predominantly consisting of surface waves generated by surf at sea and ocean shores, industrial activity, atmospheric disturbances (direct wind impact on the earth surface or indirect wind actions manifested as trees motion or building swaying), etc.. Due to influence of seismic noise, which is especially strong at low frequencies, the potentials of broad-band seismic instrumentation for recording weak seismic waves can not be realised to the full extent. It makes difficult to process records from weak earthquakes by conventional means and to provide acceptable magnitude threshold for local and regional seismic events reliably detected, located and classified.

Currently, there exists some effective approaches for reducing transient (coherent) seismic noise influence on seismic observations. One of them is based on the registering seismic waves by seismic arrays, which consists of mainly vertical single-component seismometer. It is considered to be most efficient way to noise suppression. Special mutual processing of array records which includes multichannel (group) filtering (beamforming) and spatial spectral (F-K) analysis provides a significant increase in signal-to-noise ratio (SNR) within a broad frequency band and hence estimation of main parameters of seismic phases and location of sources for weak local and regional seismic events. One can take the appliance of a seismic array to provide estimation of apparent velocities of seismic phases without a'priory knowledge on wave velocities in a medium as their main advantage [60]. The some disadvantage is a rather high cost of their installation and maintenance.

Seismological research studies based on array recordings have a long history. Large aperture arrays like LASA and NORSAR have been successfully used for a long time for investigation of different problems of regional and global seismology [14,17,29,76,60]. These arrays stimulated the development of new approaches to array data processing, particularly the new procedures based on multidimensional statistical time series analysis [18,19,37,27,68]. Nevertheless the application of new statistical procedures for large arrays shows little advantage in comparison with simple conventional beamforming and F-K analysis procedures [19]. The main reason for this is that the procedures used assumed signal and noise at different array sensors to be strongly coherent. For large aperture arrays this assumption is

not realistic (at least for the frequencies exceeding 3 Hz.) due to the lateral inhomogeneities of the Earth crust in the array site area.

In the last decade the significance of optimal statistical methods for array data processing has been recognised in connection with the deployment of small aperture arrays (SAA) such as NORESS, ARCESS, FINESA and GERESS [60,54,50,29,68,78]. The diameters of these arrays is approximately 3 km and signal wave distortions due to media inhomogeneities are not critical for signal frequencies less than 6-8 Hz. On the other hand, close location of such array sensors provides a high correlation of noise at sensors, especially in sea shore regions of this arrays deployment (Fig.1.1). The consequence here is that conventional array data processing methods like beamforming and F-K analysis, sometimes fail as a tool for noise suppression and signal parameter estimation. Finally, the progress in computer capabilities made it possible to use in an on-line mode rather sophisticated array data processing algorithms. All these circumstances justify efforts aimed at developing a new generation of statistically optimal array data processing algorithms.

The another approach for enhancing efficiency of seismic observations is the combination of array data processing techniques and polarisation 3-component data analysis that is provided by registration of seismic field with the help of 3-component seismic arrays.

Today, a vast number of digital three-component stations are deployed on a global scale. The real-time processing technique can be used for detection of seismic events, their azimuth and apparent velocity estimation, phase identification and source location using data from single 3-component station [66]. However, this seismic bulletin production technique demands the information concerning the wave velocities in a medium beneath the site. The improvement of this disadvantage of single 3-component stations and enhancement of their abilities to suppress long period seismic noise can be achieved by expansion of the seismic station to the micro array by installation of additional seismometers, located not so far from the central site.

There is another simple decision for the problem of extraction of weak seismic signal from noise and azimuth and apparent velocity estimation. It is based on the joint employment at the same observational site of different types of sensors: 3-component seismometer and horizontal strainmeters, and implementation of special algorithms for joint processing their output signals. The advantage of this method of seismic signal registration are due to simultaneous recording of distinct features of seismic field: ground surface displacement - by seismometers, and spatial derivative of this displacement - by strainmeters. Comparison of seismometer and strainmeter records gives one the information about seismic wave apparent velocity and provides the possibility to suppress transient seismic noise.

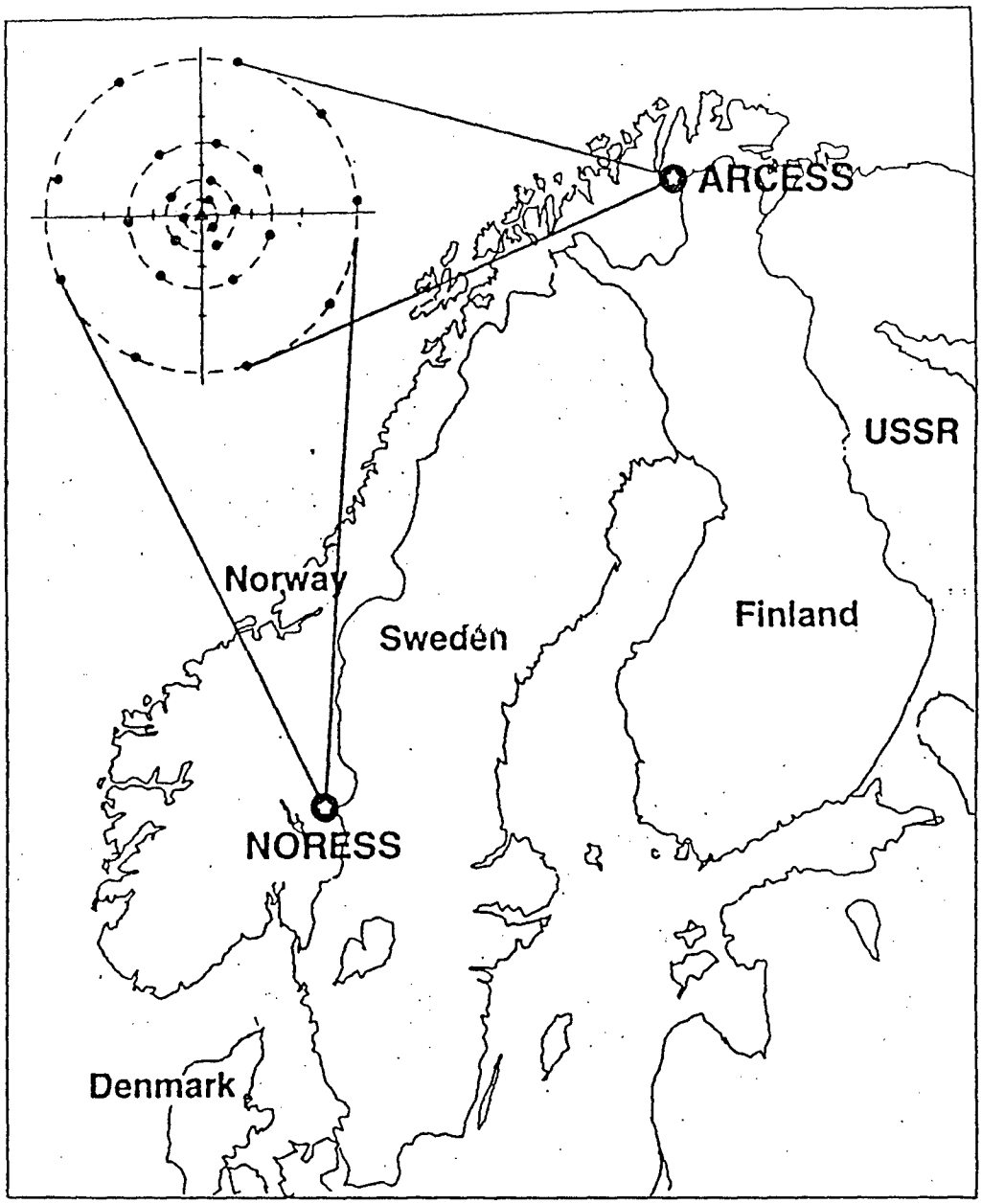


Fig. 1.1

Benioff was the first pointed out that the comparison of records of the two types instruments provides determination of wave apparent velocities using observations at a single site [7]. We should also mention paper [70] where the attempt was made to use a linear combination of outputs of strain and inertial vertical seismometers with the purpose to increase SNR in regional and teleseismic P-wave records in the situation where the noise field consists of scattered waves of the fundamental Rayleigh mode.

Only improvements in the technology of strain measurements made possible to develop a portable broad-band strainmeter [9] being suitable to realise ideas described above. As the gain of a strainmeter is proportional to an apparent slowness of a (signal or noise) seismic wave the adaptive statistical multichannel data processing technique can be used to suppress transient noise at the outputs of strain-seismometer installation and to estimate an apparent slowness of a signal wave. The similar technique have been recently rather successfully used for small aperture seismic arrays [90].

The minimal appropriate instrument configuration needed to implement this data processing technique comprises a three-component broad-band seismometer and a two-component horizontal strainmeter. We call this combination as the Strain-Inertial Micro Array (SIMA). The SIMA could be considered as the cheapest instrument able to cover the gap between a small aperture array and a single three-component digital station. SIMA being used as a tool for automated seismic bulletin production based on data from single observational site is expected to provide the advantage over 3C digital seismic station. It seems to be comparable in this sense with a micro-array while to be significantly less expensive.

Near real time seismic monitoring involves the problems of signal detection, waveform extraction and signal parameter estimation for numerous earthquakes and chemical explosions with low signal-to noise ratio (SNR) [60]. The theoretical basis of data processing in noisy environment is the statistical time series analysis [24]. Obviously, the processing algorithms must incorporate the supposed statistical properties of the seismic noise. The most of conventional seismic data processing algorithms have been designed under the simple assumption about noise, that is: a seismic noise field is a "white" Gaussian random field lacking any temporal and spatial correlation. The wide spread use of modern broad band digital seismic instruments make this assumption unacceptable because a seismic field as a rule has a strong time and spatial correlation at least partly in the frequency band covered by these instruments.

Theoretical analysis shows that significant gain in SNR can be achieved if processing algorithms take into account the correlational (spectral) features of the ambient noise [38,39,40]. Practical implementation of such statistically optimal algorithms can only be successful if they incorporate the real time dependent features of noise. Due to nonstationarity of a seismic noise field it became possible

only in the framework of an adaptive processing strategy: statistical characteristics of noise should be estimated by a special adaptation procedure involved. Such adaptation would be the most effective if it utilises the noise observations at a time interval just before an arriving signal. We call such data processing algorithms as 'adaptive statistical algorithms'. They appears to be relevant for seismic array data analysis where the noise field is generated by sea waves, industrial activities and therefore often exhibit strong coherency.

In this report a set of adaptive algorithms designed for small aperture array data processing is described. All of them incorporate the statistical estimation of a power spectral density of seismic noise, which is a matrix function in the case of multichannel data processing. Utilisation of this spectral density estimate enhances an SNR due to suppression of the coherent part of ambient seismic noise and/or interfering seismic waves.

The developed package of algorithms performs the following processing of SSA data (Fig.1.2):

a) Adaptive optimal group filtering (AOGF) of array data producing at output either undistorted signal waveforms with coloured residual noise or distorted signal waveforms with whitened residual noise.

b) Multichannel spatial rejecting filtering of array data which rejects noise and retains signal waveforms undistorted.

c) Adaptive statistically optimal detection of seismic phases.

d) Statistically optimal estimation of signal onset times.

e) Adaptive statistically optimal estimation of azimuth and apparent velocity of a signal wave obscured by strong coherent noise or an interfering wave from another seismic event.

It should be emphasised that the above algorithms are designed for the broad-band array data processing. In this regard they are strictly distinct from the technique currently used for small aperture seismic array data processing which analyses seismic data in a set of narrow frequency bands [55,61]. In our opinion the broad-band adaptive data processing technique would provide more reliable detection and more accurate parameter estimation of weak seismic phases and has some explicit computational advantages in comparison with the conventional technique.

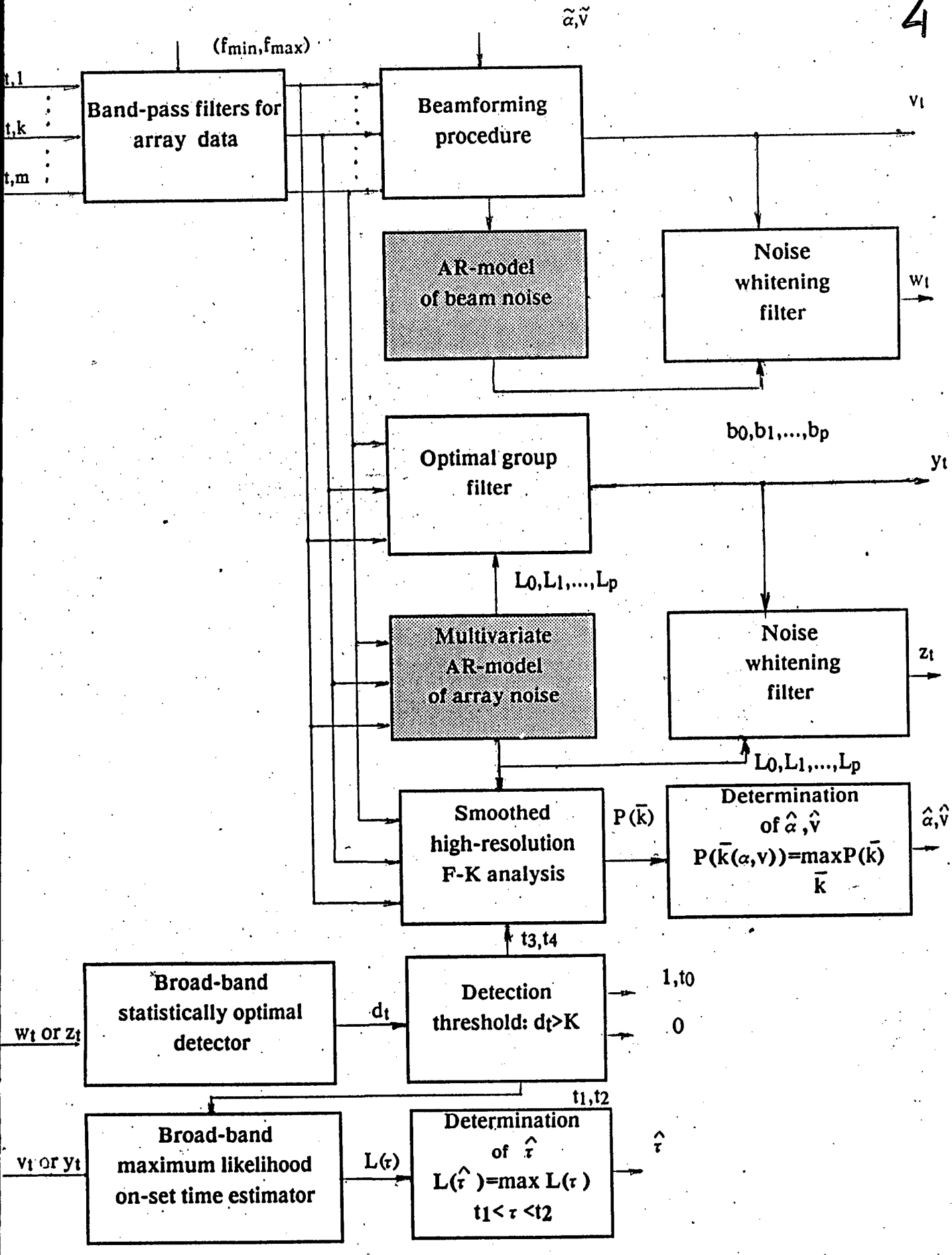

$\tilde{\alpha},\tilde{v}$
(f_{min},f_{max})
t,1
t,k
t,m
Band-pass filters for array data
Beamforming procedure
v_t
AR-model of beam noise
Noise whitening filter
w_t
$b_0,b_1,\ldots,b_p$
Optimal group filter
y_t
$L_0,L_1,\ldots,L_p$
Multivariate AR-model of array noise
Noise whitening filter
z_t
$L_0,L_1,\ldots,L_p$
Smoothed high-resolution F-K analysis
$P(\bar{k})$
Determination of $\hat{\alpha},\hat{v}$
$P(\bar{k}(\alpha,v))=\max_{\bar{k}} P(\bar{k})$
$\hat{\alpha},\hat{v}$
t_3,t_4
Broad-band statistically optimal detector
w_t or z_t
d_t
Detection threshold: $d_t>K$
1,t_0
0
t_1,t_2
Broad-band maximum likelihood on-set time estimator
v_t or y_t
$L(\tau)$
Determination of $\hat{\tau}$
$L(\hat{\tau})=\max L(\tau)$
$t_1<\tau<t_2$
$\hat{\tau}$

Fig. 1.2.

2. Mathematical models of multi-component seismic observations and possible formulations of problems of their interpretations

Let us suppose hereinafter that seismic waves are generated by remote seismic source and each of the seismic phases (*P, S, L, R* etc.) is a plain wave. It is supposed as well that the body wave phases arrive from the homogeneous lower half space on a surface batch of laterally homogeneous layers in accordance with unit steering vectors $a_W=(a_{Wx}, a_{Wy}, a_{Wz})^T$, where w represents the wave-type index. We designate as v_W a velocity of a w-type wave in the half space directly beneath the batch of layers. Let us assume the origin of coordinates to be setteled on the surface of the half space, with the *Z*-axis directed down, *Y*-axis to the North and *X*-axis to the East; the wave azimuth α be counted clockwise from the positive direction of the *Y* axis, and the wave incidence angle β_W - from the positive direction of the *Z* axis. Then the vector a_W can be written as $a_W=(\sin\alpha \sin\beta_W , \cos\alpha \sin\beta_W , \cos\beta_W)^T$ (T is the sign of transposition). For surface waves $\beta_W = \pi/2$ *and* $\cos\beta_W = 0$.

A medium displacement $w(t,r)=(w_x(t,r), w_y(t,r), w_z(t,r))^T$ for a particular seismic phase at an arbitrary point $r = (r_x, r_y, r_z)^T$ of the homogeneous half space can be expressed as follows:

$$w(t,r) = s_W (t - (r^T a_W)/ v_W) b_W \qquad (1)$$

where $s_W(t)$ is a waveform of a seismic phase at the origin of coordinates, $b_W = (b_{Wx} , b_{Wy} , b_{Wz})^T$ - is a unit vector of seismic phase oscillations which is determined by a wave the incidence angle β and azimuth α. In frequency domain eq. (1) has the form

$$w(f,r) = s_W (f) \exp[-i2\pi f(r^T a_W)/v_W] b_W \qquad (2)$$

where $s_W(f)$ is the complex spectrum of a phase waveform. For different types of seismic waves the vectors b_W are expressed by the following equations:

for P-waves
$$b_P = \begin{bmatrix} \sin\alpha \sin\beta \\ \cos\alpha \sin\beta \\ \cos\beta \end{bmatrix} \qquad (3a)$$

for SH-waves and for Love waves $$b_L = \begin{bmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{bmatrix}; \quad (3b)$$

for SV-waves $$b_V = \begin{bmatrix} \sin\alpha\cos\beta \\ \cos\alpha\cos\beta \\ \sin\beta \end{bmatrix}; \quad (3c)$$

for Rayleigh waves $$b_R = \begin{bmatrix} i\sin\alpha\sin\psi \\ i\cos\alpha\sin\psi \\ \cos\psi \end{bmatrix}; \quad (3d)$$

where $\psi = arctg\ (e)$, e represents the ellipse factor of a Rayleigh wave, e.g. the ratio of the small axis of a polarisation ellipse to the large one; $i=\sqrt{-1}$ characterises the phase shift of $\pi/2$ between the vertical and horizontal components of Rayleigh wave displacements.

Let us introduce the 3-dimensional vector of a apparent slownesses: $p_W = (p_x , p_y , p_z)^T = a / v_W$. Then the wave field eq.(2) can be expressed as a function of the vector p_W as follows:

$$w(f,r) = s_W(f)\ exp[-i2\pi f(r^T p_W)]\ b(p_W) \quad (4)$$

Eq.(4) is based on the following simple relations:

$$sin\alpha = p_x / p_h, \quad cos\alpha = p_y / p_h , \quad sin\beta_W = p_h v_W ; \quad p_h = (p^2_x + p^2_y)^{-1/2} \quad (5)$$

Equations (3) and (5) imply that

for P-waves $$b_P = \begin{bmatrix} p_x \\ p_y \\ \sqrt{v_V^{-2} - p_h^2} \end{bmatrix} . v_P \begin{bmatrix} p_y \\ p_x \\ 0 \end{bmatrix} . p_h^{-1} \quad (6a)$$

for SH-waves and Love waves $$b_L = \begin{bmatrix} p_y \\ p_x \\ 0 \end{bmatrix} . p_h^{-1} \quad (6b)$$

for SV - waves $$b_V = \begin{bmatrix} p_x \\ p_y \\ p_h^2 \Big/ \sqrt{v_V^{-2} - p_h^2} \end{bmatrix} \sqrt{p_h^{-2} - v_V^2} \quad (6c)$$

for Rayleigh waves $$b_R = \begin{bmatrix} & p_x & \\ & p_y & \\ -i & ctg\psi & p_h \end{bmatrix} ip_h^{-1} \sin\psi \qquad (6d)$$

Note that for Rayleigh and Love waves values of p_x and p_y depend on frequency f.

If one neglects the effects of wave reflection from layer borders and conversion of wave types on borders while plane waves propagates through layered medium and be confined to taking into account only the effects of wave refraction (the ray propagation approximation), then according to the Snellius law values of $p_h = sin\beta_W / v_W$ at any point within batch of layers (and beneath the day surface) is constant and equal to its value in the half space. As $p_x = sin\alpha\, p_h$ and $p_y = cos\alpha\, p_h$, values of the 2-dimensional vector $p = (p_x, p_y)^T$ of horizontal apparent slowness are too constant in all medium points. This vector value is the ray parameter and determines the propagation path of a seismic ray in such medium [1]. Based upon eq.(5) and eq.(6) one can come to conclusion that a wave seismic field in any point r of a laterally homogeneous medium is determined only by the vector p and wave velocity v_W in that layer where the point r is located and is independent from other parameters of layers. Thus, eq.(4) initially written for the homogeneous space is valid for any point of a laterally homogeneous (layered) medium and in particular for points beneath the day surface.

Considering eq.(4) enables us to interpret the field $w(f,r)$ on the daysurface, e.g. for $r = u=(x, y, 0)$ as a result of a signal $s_W(f)$ propagation through a linear system:

$$w(f,u) = g_W(f,u,p)s_W(f). \qquad (7)$$

The frequency response of this system (in a case of flat-stratified medium) is

$$g_W(f,u,p) = exp[-i2\pi f(u^T p)]\; b(p,v_W) \qquad (8)$$

If a batch of layers is fine-stratified and contrasting enough so it is impossible to neglect the effects of wave reflection and wave type transformation, then a field $w(f,u)$ on the day surface admits yet the representation by eq.(7). However, in this case the frequency response of the corresponding linear system can not be expressed by such simple formula as eq.(8). Nevertheless for an arbitrary laterally homogeneous batch it can be determined with the help of rather effective computational methods like the ones of Thompson-Haskel or Kennet [34].

Let us consider a system of seismic observations consisting of m 3-component seismometers registering displacements of a medium and located at the day surface at points u_i. If distances between points u_i are rather small so the medium can be considered permanent within the aperture of the system then such system of seismic observations can be referred as 3-component array. For this case the set of complex spectra of seismometer outputs - the $3m$-dimensional column vector $y_W(f)=(y_{Wi}(f), i=\overline{1,3m}$) - may be expressed as

$$y_W(f) = h_W(f,p)\, s_W(f) \qquad (9)$$

where $h_W(f,p)=(g_W(f,u_i,p),\ i=\overline{1,3m}\ )$ is the $3m$-dimensional column vector of frequency responses of linear systems that are determined by propagation paths of the plain wave from the lower half space to seismometers outputs. In particular for ray approximation one can write

$$h_W(f,s) = (exp[-i2\pi f\,(u^T{}_i\ ,p_W)]\ b_i(p,v_W),\ \ i=\overline{1,m}) \qquad (10)$$

If all seismometers within the system of observations are 3-component ones, then vectors $b_i(p,v_W)$ in eq.(10) are the same for all i. The dependence b_i from i in eq.(10) takes into account that for some array seismometers some component instruments may be missed. In such case the corresponding components of vectors $b_i(p,v_W)$ must be assumed equal to zero.

The considerations made before allow to conclude that in the framework of ray propagation approximation without any loss of generality the origin of coordinates may be placed at u_1 - the point of location of the central array sensor. Then $s_W(f)$ can be interpreted as the waveform of a seismic phase at the central sensor.

Eq.(9), (10) demonstrate that in the framework of ray propagation approach signals from an array of 3-component seismometers contain information on vector p of seismic phase apparent slownesses, wave velocity v_W in the layer beneath the surface and phase waveform $s_W(f)$. It is worthy to be mentioned that information on apparent slowness vector p is contained in relative time delays $\tau_k= (u_k- u_1)^T p$ of sensor signals and in polarisation characteristics of each 3-component seismometer output (expressed by vectors $b_k(p,v_W)$). It is very important that relative delays τ_k do not depend on velocity v_W of a seismic wave phase in the layer where seismometers are placed. In the contrast, the polarisation characteristics are strongly dependent on v_W. As a rule, the upper subsurface layer has small phase velocities, often known with a high uncertainty and even likely changing due to atmosphere

conditions. In this conditions it occurs that angles of incidence β_W of body seismic waves at the day surface might be small. As result the accuracy of β_W measurement based on polarisation characteristics is rather small too [53]. Hence more accurate methods to measure apparent slowness p are those based on time delays τ_k , for example, methods of the spatial spectral analysis (F-K analysis).

As it follows from eq.(3) and eq.(6) an amplitude ratio between signals of seismometer horizontal components is determined by a seismic wave azimuth (which is normally the same for all seismic phases). Consequently 3-component array observations submit some information about wave azimuth additional to the relative time delays τ_k, the information is comprised in wave polarisation characteristics and do not depend on wave velocity v_W in the surface layer.

If vector p is determined a phase velocity v_W can be estimated from polarisation characteristics of three component observations based on eq.(6). And at last, if p and v_W are known there appear the good conditions for determination of seismic phase waveform $s_W(f)$ $(s_W(t))$ using multidimensional observations by eq.(9), (10), e.g. extraction of this function from a noise background. This is quite substantial in case of small signal-to-noise ratio.

In seismic monitoring measurements the apparent slowness vector p determination appears to be crucial for effective location of seismic event epicentres using data from single seismic array. The determination of seismic phase waveforms $s_W(f)$ is necessary for identification of a seismic source type and estimation of a sourse seismic moment tensor. The estimation of wave velocity v_W is important for investigation of regional medium structure on the basis of seismic data.

If seismic observations are performed at platform regions, where seismometers should be installed on a batch of sedimentary layers the ray propagation approximation of a medium frequency response $h_W(f,p)$ by eq.(10) is appeared to be of nearly to use. Under this condition the problem of determination of apparent slowness vectors p and waveforms $s_W(f)$ of seismic phases in the lower half space should be solved by some ways completely different from traditional ones [53]. For example, the conventional F-K analysis is practically of no use here due to wave reverberation (numerous wave reflections and transformations of wave types). Really, in this case a maximum of a spatial spectrum of a seismic field at the day surface (which should be related with the main refraction wave (having the same apparent slowness p as an original wave in the half space) usually is not high enough and does not provide the reliable estimation of vector p.

If a model of a surface layer batch can be considered known, the appropriate approach comprises (roughly speaking) comparison of array observations with synthetic seismograms calculated for different values of p. In result, the procedure

of vector p estimation is reduced to the minimisation in p of certain functional from observations which depends from the medium frequency response $h_W(f,p)$ [39]:

$$\hat{p} = arg \min_{p} \Phi [y(f), h_W(f,p)] \qquad (11)$$

If a medium model is known with an accuracy up to a vector of nuisance parameters υ (υ for instance, can be thickness of layers and velocities of waves in layers) it appears possible (in case of sufficiently large number of stations in an array) to determine the vector p altogether with the vector υ by minimising the functional eq.(11) in the set of the all unknown parameters [39]:

$$\begin{bmatrix} \hat{p} \\ \hat{\vartheta} \end{bmatrix} = arg \min_{p,\vartheta} \Phi [y(f), h_W(f,p,\vartheta)] \qquad (12)$$

And at last, for the surface waves the apparent slowness p depends on a frequency: $p=p(f)$, in accordance with the medium dispersion curve, unambiguously connected with parameters of the laterally homogeneous medium. The determination of a function $p(f)$ from 3-component array data eq.(9), (10) can be performed either by means of the traditional F-K analysis or (if noisy data) by more complicated and more accurate statistical procedures [39].

The described above approaches for determination of main seismic phase characteristics seem to be mostly applicable to the small aperture (up to 5 km) and medium aperture (up to 20 km) seismic arrays, composed of dozens of seismometers. The well known European arrays like NORESS, ARCESS, FINESA, GERESS are the examples of small aperture arrays. Apertures of these arrays allow to neglect lateral heterogeneity of medium beneath an array, which makes the above discussed models of seismic observations acceptable. At the same time, the sufficiently high coherency of seismic noise in array sensors ensure the provision of high accuracy of seismic phase parameter estimation and waveforms restoration from the noise background. This can be achieved by implementation of statistically optimal methods for data processing in particularly, those described in the following sections.

Note that all the indicated arrays consist both of three component and single component vertical seismometers, the latter are much numerous. (To our consideration this has been made to reduce the cost of such arrays installation and maintenance). Should one needs to take into account an absence of some components of array seismometers it is enough to assume the corresponding components of vectors $b_k(p,v)$ in the models of observations by eq.(9), (10) equal to zero.

Recently the systems of seismic observations based on the micro-arrays have acquired significant popularity [46]. The latter are arrays consisting of small number (4-10) of sensors, located at distances of several hundred meters. Such installations are distinguished for the small cost, due to the cheapness of the signal transmission from sensors to the data acquisition centre. However, in some cases it is rather difficult to utilise the high coherency of signals from different microarray sensors because the relative time delays τ_k of these signals are too small in comparison with the typical periods of seismic signals being processed. Later yields the accuracy of measurements of delays τ_k to be rather bad in a case of small signal-to-noise ratio. In such case the known theoretical advantage concerning an information extraction from the relative signal time delays in comparison with its extraction from the 3-component signal amplitude relations [26] appears not valid. Therefore, it might be expedient for microarrays to use a discussed below alternative method for apparent slowness vector estimation, which in case of ray approximation (models of observations by eq.(9), (10)) also do not need an information on a wave velocity v_W in the surface layer.

Let us assign the vector of differences between signals of k-th three component sensor and 1-st (the central) three component sensor as $d_W(f) = (d_{Wk}(f), k=\overline{1,m})$. In accordance with eq.(9),(10)

$$d_{Wk}(f) = exp[-i2\pi f(u^T_k p) - exp[-i2\pi f(u^T_1 p)]\ b(p,v_W)\ s_W(f) \qquad (13)$$

When distances $l_k = u_k - u_1$ are small the eq.(13) can be approximately written as:

$$d_{Wk}(f) = (l^T_k p)exp[-i2\pi f(u^T_k p)]\ b(p,v_W)[-2\pi f i s_W(f)] \qquad (14)$$

Thus, the difference between j-th components of spectra of k-th and 1-st sensor divided by the j-th component of a spectrum of k-th sensor is equal to:

$$d_{Wkj}(f) / y_{Wkj}(f) = -2\pi i(l^T_k p), \quad k=\overline{1,m}, \quad j=\overline{1,3} \qquad (15)$$

So the apparent slownesses vector p can be obtained (at every f) by solving the system of 3m linear equations. It is clear, that this procedure is much more simple for calculations, than determination of the vector p from the relative time delays τ_k of sensor signals. Let us remark, that

a) this procedure is the same for all wave phases in contrast with procedures of p determination from the polarisation characteristics;

b) for body waves the vector p estimations obtained for different frequencies f can be then averaged through f for a frequency band of interest;

c) for surface waves this procedure, done for each frequency f allows to determine the function $p(f)$, e.g. the dispersion curve of a surface wave considered.

As $[-2\pi f s_W(f)]$ is the Fourjer transform of the time derivative $ds_W(t)/dt$ of a seismic phase waveform, one easily derive from (14) that for body waves (where vector p is independent of frequency) the difference $d_{Wkj}(t)$ between j-th components of k-th and 1-st sensor records is related with the record $y_{Wkj}(t)$ of j-th component of k-th sensor by the following simple expression:

$$d_{Wkj}(t) = -(l^T{}_{kl} p)\, dy_{Wkj}(t)/dt\,, \quad k=\overline{1,m}, \quad j=\overline{1,3} \qquad (16)$$

The system of linear equations (16) have to be solved for each t and then the results have to be averaged in t within an interval of seismic phase existence. This allows one to determine vector p using calculations in the time domain. The majority of modern broad band seismometers are velocimeters, e.g. they measure the velocity of a medium seismic displacement $v_W(t) = dy_{Wkj}(t)/dt$. Therefore to implement the described procedure of apparent slowness vector estimation one has to preliminary integrate the velocimeter records over t with the purpose to calculate the differences $d_{Wk}(t)$. The integration provides the additional suppression of high frequency noise.

Mentioned in the Introduction the strain-seismometer micro array (SIMA) is the simplest instrument for seismic observations which provides the possibility of apparent slowness vector determination independently on an information about a wave velocity in a media layer beneath the surface. The described above p-estimation procedures can be easily implemented to SIMA data. A SIMA consists of one 3-component seismometer and two horizontal strainmeters with a several meters base length L installed close to the seismometer in the ortoghonal directions and registering medium strain in the seismic frequency range. An output signal of a strainmeter is proportional to the difference of medium displacements at the ends of the strainmeter base. Therefore in sense of information provision a SIMA is equivalent to the combination of a central 3-component seismometer and two 1-component horizontal seismometers, steered to the ortoghonal directions and remote from the central sensor at the distance L. The vector p estimate can be gained from a system of two linear equations (15) or (16) where indexes (k,j) are: $(k=1,j=1)$, $(k=2,j=2)$. The determination of wave velocity v_W from SIMA data can be made using polarisation characteristics of 3-component seismometer output. After an apparent slownesses vector p and a wave phase velocity v_W are determined a

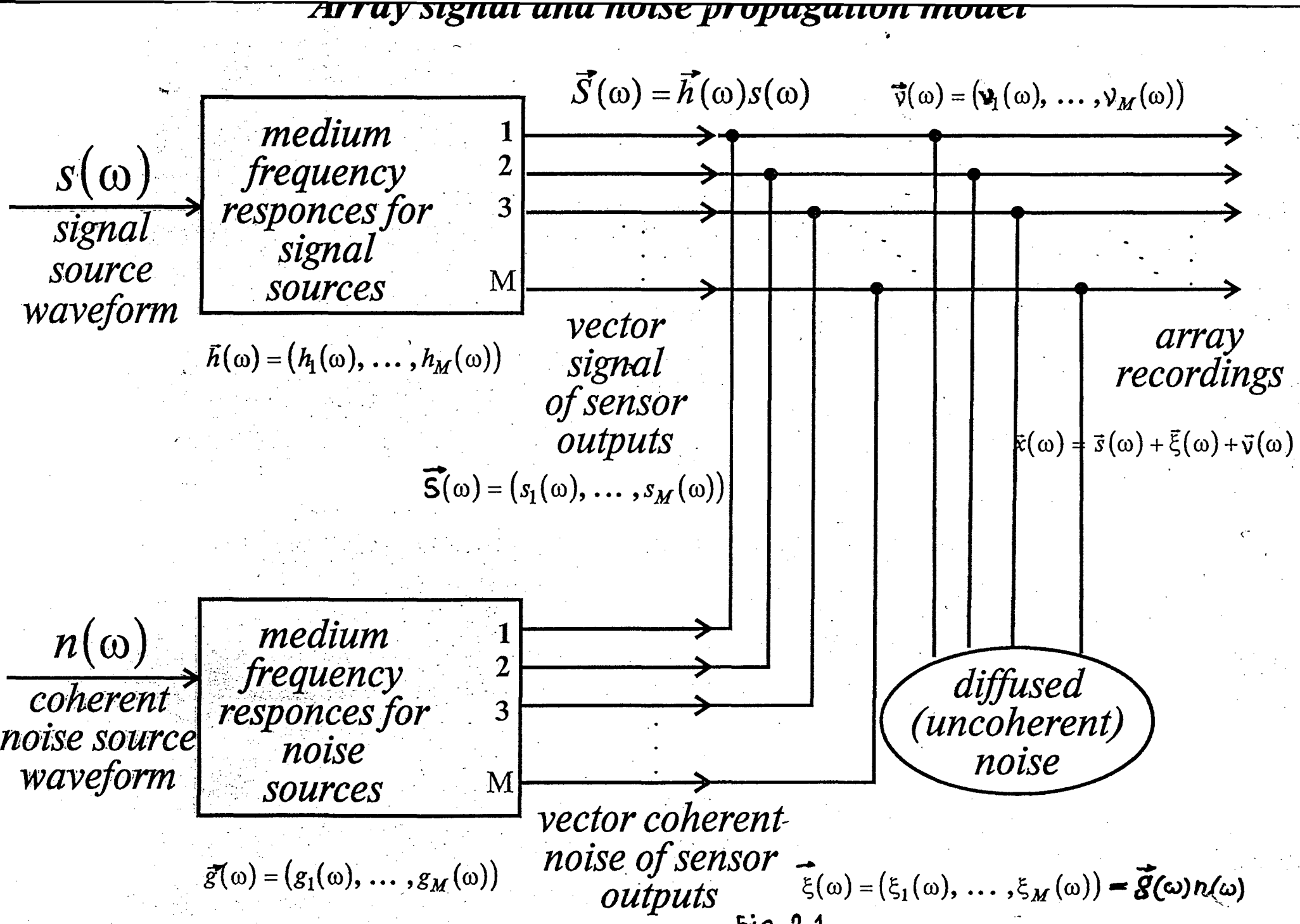

Array signal and noise propagation model
$\vec{S}(\omega) = \vec{h}(\omega)s(\omega)$
$\vec{v}(\omega) = (v_1(\omega), \ldots, v_M(\omega))$
$s(\omega)$
signal source waveform
medium frequency responces for signal sources
1
2
3
M
$\vec{h}(\omega) = (h_1(\omega), \ldots, h_M(\omega))$
vector signal of sensor outputs
$\vec{S}(\omega) = (s_1(\omega), \ldots, s_M(\omega))$
array recordings
$\vec{x}(\omega) = \vec{s}(\omega) + \vec{\xi}(\omega) + \vec{v}(\omega)$
$n(\omega)$
coherent noise source waveform
medium frequency responces for noise sources
1
2
3
M
diffused (uncoherent) noise
$\vec{g}(\omega) = (g_1(\omega), \ldots, g_M(\omega))$
vector coherent noise of sensor outputs
$\vec{\xi}(\omega) = (\xi_1(\omega), \ldots, \xi_M(\omega)) = \vec{g}(\omega)n(\omega)$

Fig. 2.1.

seismic phase waveform $s_W(t)$ can be estimated from SIMA 5-component records with the help of a noise suppressing procedure, described hereafter in Section 3.

Signals of seismometers (and strainmeters) are usually observing against a noise background $\xi(t) = (\xi_{kj}(t),\ k = \overline{1,m};\ j = \overline{1,3})$. Therefore output signals of a small aperture system of seismic observations compose a multidimensional random time series

$$x_W(t) = y_W(t) + \xi(t) = h_W(t,p) * s_W(t) + \xi(t) \qquad (17)$$

where $g_W(t,p)$ is a vector impulse response of a medium for given wave phase, (equal to Fourjer transform of a vector-function (10)) * is the sign of convolution.. In the frequency domain eq. (17) can be written as

$$x_W(f) = h_W(f,p)\, s_W(f) + \xi\,(f) \qquad (18)$$

where $\xi\,(f)$ is a column 3m dimensional vector of a noise complex spectrum.

The noise presence in observations results in the necessity to take into account distorting effects of noise while resolving the discussed above problems of seismic phase parameter determination. E.g. these problems should be formulated as problems of statistical time series analysis. Theory of optimal statistical inferences [13,24] should be implemented for their solution. It meets some difficulties because this theory has been mostly developed for case of known statistical characteristics of noise. For seismic noise it is usually not so.

Numerous investigations have brought enough proves for seismic noise $\xi(t)$ to be considered as a Gaussian process, which may be assumed to be stationary (at a time intervals typical for seismic signal durations) with zero mean and the smooth matrix power spectrum density (MPSD)) = $E\{\xi(f)\xi^T(f)\}$. Here E represents the sign of mathematical expectation. A MPSD $\boldsymbol{F}(f)$ reflects seismic noise space and time correlations, which are as a rule non-zero and mostly developed in a low-frequency band. It is especially typical for seismic noise in regions close to sea and ocean shores where seismic noise are generated mostly by a surf. In such regions the space correlation of noise (coherence of noise) is so high that for frequencies lower than 1-3 Hz the MPSD $\boldsymbol{F}(f)$ of a noise vector time series $\xi(t)$ registered by a small aperture seismic array is nearby to singular [20,42]. The proximity of a determinant of $\boldsymbol{F}(f)$ to zero forms the criteria for noise coherency in regard to a given observational system.

As it follows from general points of mathematical statistics if a determinant of $\boldsymbol{F}(f)$ is close to zero the probabilities of errors for statistically optimal decision rules are close to zero as well. Note that for non-optimal procedures, it is by no means

necessary. Hence it seems to be rather desirable to take into account an information on noise MPSD $\boldsymbol{F}(f)$ for synthesis of statistically optimal algorithms for processing of multi-component seismic observations. It often provides a significant increase in accuracy of seismic signals parameter estimation due to the effect of "compensation" (suppression) of noise by statistically optimal data processing procedures .

In order to achieve the theoretical efficiency of noise suppression the strict information about its MPSD $\boldsymbol{F}(f)$ is demanded. But for long time intervals a noise time series $\xi(t)$ can not be regarded as a multidimensional stationary process. It is so, because its MPSD $\boldsymbol{F}(f)$ being defined by current characteristics of a seismic noise field, is changing in time. Therefore the statistically optimal procedures of multi-component seismic data analysis can be efficient only in the framework of adaptive approach where the noise MPSD is continuously or periodically estimated using current noise observations.

3. Statistically optimal extraction of seismic signals from noise

3.1 Statistically optimal group filter (Winner filter)

According to eq.(2.18), to restore a waveform $s(t)$ $(s(f))$ of a seismic wave from data $x(t)$ recorded by a 3-component seismic array, one should make up for a distorting effect of the medium and clear the signal off the noise. As it follows from statistical theory of time series analysis, if distribution of noises $\xi(f)$ is Gaussian the best procedure for restoring the function $s(f)$ is a linear group filtering, according to which the scalar complex estimation $\hat{s}(f)$ of a waveform $s(f)$ can be found as follows:

$$\hat{s}(f) = \phi^*(f)\, x(f) \qquad (1)$$

where $\phi(f) = (\phi_{jk}(f),\ j = \overline{1,m},\ k = \overline{1,3})$ is a 3m-component vector frequency response of a group filter, symbol * designate Hermitian conjugation.

The problem of optimal estimation of a waveform $s(f)$ (the optimal Winner filtering) consists in seeking for such frequency response which would minimise the square mean deviation

$$E_S \{ [\hat{s}(f) - s(f)]^2 \} = min \qquad (2)$$

under the condition of unbiased signal estimation:

$$E_S \{ \hat{s}(f) \} = s(f) \qquad (3)$$

In the above equations E_S is the sign of conditional mathematical expectation under the condition that signal is given; the eq.(2),(3) must be valid for each frequency f for any signal realisation. Requirements by eq.(2),(3) are equivalent to the requirement on signal estimate dispersion to be minimal:

$$D_S \{ \hat{s}(f) \} = min \qquad (4)$$

With the natural assumption that a random signal $s(t)$ has zero mean and is statistically independent with noises $\xi(t)$, condition eq.(3) leads to the following constraint on the frequency response of an Winner filter (optimal group filter, OGF):

$$\phi_O^*(f)\, h_W(f,p) = 1 \qquad (5)$$

Note that condition eq.(3) for an output of the Winner filter to be unbiased is reasonable only for group filtering, where a scalar waveform $s(f)$ is restored from a multidimensional time series of kind eq.(2.18) with $m > 1$. In a single- dimensional case, the Wiener filtering by eq.(1), (2) unavoidably distorts a signal that makes condition eq.(3) impossible.

The minimisation of dispersion of signal estimate eq.(4) under condition eq.(5) results in the following expression for the frequency response of the OGF [19,68,38,20] :

$$\phi^*_O(f) = [h^*_W(f,p) \boldsymbol{F}^{-1}(f)] \,/\, [h^*_W(f,p)\, \boldsymbol{F}^{-1}(f)\, h_W(f,p)] \qquad (6)$$

where $\boldsymbol{F}^{-1}(f)$ is a inverse matrix spectral density of noise $\xi(t)$; $\boldsymbol{F}(f) = E\,[\xi(f)\xi^*(f)]$. If the vector function $h_W(f,p)$ in eq.(6) corresponds to the frequency response of a medium for the given seismic phase propagation, the OGF output signal in the no noise condition coincides with a phase waveform $s_W(f)$. Indeed, in the absence of noise $(\xi(t) = 0)$

$$\hat{s}(f) = \{[h^*_W(f,p)\, \boldsymbol{F}^{-1}(f)] \,/\, [h^*_W(f,p)\, \boldsymbol{F}^{-1}(f)\, h_W(f,p)]\}\, h_W(f,p)\, s_W(f) = s_W(f). \qquad (7)$$

For the ray approximation of seismic wave propagation $h_W(f,p)$ is expressed by eq.(2.10) and depends on a velocity v_W of a seismic wave in the near-surface layer: $h_W(f,p) = h(f,p,v_W)$. It is easy to show that the noise power spectral density $\eta(f)$ at the output of OGF is equal to

$$q_\eta(f) = [h^*_W(f,p)\, \boldsymbol{F}^{-1}(f)\, h_W(f,p)]^{-1} \qquad (8)$$

If noise $\xi(t)$ consist of only non-correlated diffusion seismic noise, then $\boldsymbol{F}^{-1}(f) = \sigma^{-2}\, \boldsymbol{D}(f)$ e.g. is a diagonal matrix. In this case, it is often (and groundlessly) assumed that diffusion noise is white. Then $\boldsymbol{D}(t) = \boldsymbol{I}$ and the optimal group filtering procedure coincides with the classical beamforming procedure.

The optimal group filter eq.(6) does not distort a signal waveform only when its vector-function $h_W(f)$ coincides with the true frequency response of the medium for the signal propagation paths. In particular, for a plain wave and the ray propagation approach we have: $h_W(f) = h(f,p,v_W)$, where the vector p is to coincide with the true apparent slowness vector of the signal wave and v_W - with the true wave velocity in the medium. These values should be preliminary estimated with the help of procedures described in Section 4 below. An other possible approach consists in scanning the areas expected signal arrival directions (and/or wave velocities) with the help of a "fan" of filters eq.(6) with different functions $h(f,p,v_W)$. The values of p and v_W on which the maximum power of some filter output is attained can serve as the seismic wave parameter estimates. The OGF output signal for this direction is a signal waveform estimate.

It is well known [77] that a statistically optimal detector of random signals should incorporate a noise whitening filter. If signals are detected using array recordings the noise whitening filter and detection procedures should follow the BF or OGF noise suppression procedures. The scalar noise whitening filter being used after BF must have frequency response (FR) $r_W(f,p)= [h^*_W(f,p)\boldsymbol{F}(f)h_W(f,p)]^{-1/2}$ and application of such filter after the OGF have to be made with the frequency response $u_W(f,p)= h^*_W(f,p)\boldsymbol{F}^{-1}(f)h\,(f,p)]^{1/2}$. In practice it is more convenient to use the optimal group filter in a form of sequence of the group filter with VFR $k_W(f,p)=h^*_W(f,p)\boldsymbol{F}^{-1}(f)$ followed by the two scalar filters with FR $u_W^{-1/2}(f,p)$ and $[u_W^{-1/2}(f,p)]^*$. The first scalar filter produces a time series with whitened noise, the second - a time series with an undistorted signal waveform.

If the matrix function $\boldsymbol{F}(f)$ used for calculation of the OGF frequency response via equation (6) is the same as for the real MPSD of current array noise then the OGF provides an output with significantly greater then for BF SNR. Theoretically the OGF SNR gain tends to infinity if a determinant of $\boldsymbol{F}(f)$ tends to zero. This occurs in the case of purely coherent noise. The determinant of $\boldsymbol{F}(f)$ thus can serves as a measure of noise coherency and indicates situations where implementation of the OGF would be successful. Noise studies based on Scandinavian small aperture array records have demonstrated that due to close sensor locations the array noise for this region has the nearly singular MPSD for frequencies from 0.1 up to 1-3 Hz.

3.2 Adaptation of optimal group filter to variations of noise spectrum.

Wide spread applications of procedures recommended by statistical time series analysis such as the OGF is restricted by two circumstances. Firstly, their use in the on-line mode requires rather powerful computer. Secondly, the precise information about current noise MPSD is required for an efficient coherent noise suppression.

However, seismic noise is usually not stationary. So the OGF can be effective only in an adaptive processing system, where a MPSD $\boldsymbol{F}(f)$ is periodically estimated (updated) using current noise observations. We call such array data processing procedure as the adaptive optimal group filtering (AOGF).

Classical non-parametric methods of multidimensional spectral analysis do not provide a good estimation of $\boldsymbol{F}(f)$ if a noise MPSD is close to being singular, especially if a number of array sensors is large. For this reason past applications of different versions of adaptive optimal group filtering procedures have not been too successful.

We have found that the AOGF procedure with group filter frequency response by eq.(6) in most cases has a much better performance than the conventional beamforming (BF) procedure if a noise MPSD is estimated with the multidimensional autoregressive moving average (ARMA) modelling of noise records [38]. It means that an inverse MPSD is evaluated in the form of the matrix rational function [24]:

$$\hat{F}^{-1}(f) = \Big(\sum_{k=0}^{p} A_k\, e^{-i\,2\pi f k\Delta}\Big)\Big(\sum_{l=-q}^{q} L_l\, e^{-i\,2\pi f l\Delta}\Big)^{-1}\Big(\sum_{k=0}^{p} A_k^T\, e^{i\,2\pi f k\Delta}\Big) \quad (9)$$

where Δ is the sampling interval, matrices $\boldsymbol{A}_k$, $k \in 0,\ldots,p$, are determined using first $p+1$ sample matrix autocorrelations of noise observations with the help of a computationally effective multichannel version of the Levinson-Durbin procedure [82,35,38]. Matrix MA-coefficients $\boldsymbol{L}_l$ are calculated as weighted autcovariance matrices with lags $l \in -q,\ldots,q$ of a time series which is produced from noise data by the multichannel whitening filtering using matrix AR-coefficients $\boldsymbol{A}_k$

If in accordance with the concept of adaptive statistical processing of experimental data, a MPSD $\boldsymbol{F}(f)$ have to be determined from observations, the two situations can be faced:

a) array records comprises an interval where only "pure" noise is present;

b) all observations contain a mixture of a signal and noise.

In the first case, an estimate of MPSD $\boldsymbol{F}(f)$ can be obtained from observations of a "pure" noise (in particular, with the help of a ARMA-modeling). In the second case, this way seems at a first sight impossible since the MPSD of observations consists of a seismic signal and noise and is equal

$$\boldsymbol{F}_{xw}(f) = \boldsymbol{F}(f) + h_w(f)h^*_w(f)\nu(f), \quad (10)$$

where $\nu(f)$ is a power spectral density of a seismic phase waveform. It is the function by eq. (10), that really will be estimated if a signal and noise mixture is used. The estimate with the MPSD $\boldsymbol{F}_{xw}(f)$ obtained in this case is often largely

different from the noise MPSD $\boldsymbol{F}(f)$, especially if a signal-to-noise ratio is large enough.

The following property of the OGF by eq.(6) (which can be called as the "adaptation stability") presents a some theoretical advantage of the OGF: it is easy to show [39,45] that substitution of $\boldsymbol{F}_{xw}(f)$ by eq.(10) in eq.(6) instead $\boldsymbol{F}(f)$ does not change the value of the OGF frequency response $\phi_O(f)$. To prove this one should use for inversion of the matrix $\boldsymbol{F}_{xw}(f)$ the well known Bartlett formula [13]. It is obvious, however, that this theoretical advantage is meaningful in practice only if an estimate of $\boldsymbol{F}_{xw}(f)$ is accurate enough (and the vector medium frequency response for a signal wave $h_w(f)$ being used in eq.(6) corresponds to the true one). Some experimental results were obtained from NORESS array data concerning adaptation of the OGF using "pure" noise and a signal and noise mixture. This results are discussed in Section 9 below.

3.3 Spatial Rejecting Group Filter

The best noise suppression using group filtering procedures can be achieved if noise is close to coherent. This property usually posseses transient noise generated by one or several sources localised in a medium. Real seismic noise consists normally of such noise superposing with diffusion microseismic noise caused by a huge number of independent sources. I.e. a real noise process at array sensor outputs can be written down as follows:

$$\xi(t)=\sum_{k=1}^{s} q_k(t) * \zeta_k(t) + \delta(t), \qquad (11)$$

where $\zeta_k(t)$ is a scalar noise process in k- th source of coherent noise; $q_k(t)$ is a $3m$-dimensional vector of impulse transfer functions of noise propagation paths through the medium from k-th noise source to array sensors; s is a number of coherent noise sources; $\delta(t)$ - is a vector diffusion noise process, * is the sign of convolution.

If processes $\zeta_k(t)$, $k = \overline{1,s}$ and $\delta(t)$ are mutually non-correlated, the matrix power spectral density of process $\xi(t)$ has the following form [38,40]:

$$\boldsymbol{F}(\omega) = \sum_{k=1}^{s} q_k(f) J_k(f)\, q_k^{*}(f) + \sigma^2 \boldsymbol{D}(f) = \boldsymbol{R}_S(f) + \sigma^2 \boldsymbol{D}(f) \qquad (12)$$

where

$$\boldsymbol{R}_S = \boldsymbol{Q}_S(f)\, \Lambda_S(f); \qquad \boldsymbol{Q}_S(f) = [q_1(f),.....,q_S(f)];$$

$$\Lambda_S(f)=\begin{bmatrix} j_1(f) & & 0 \\ . & . & . \\ 0 & & j_S(f) \end{bmatrix}$$

where - $J_k(f)$ is a power spectral density of process $\zeta_k(t)$; $\boldsymbol{D}(f)$ - is a matrix power spectral density of diffusion noise $\delta(t)$; σ^2 - is a diffusion noise power (for simplicity's sake we consider it to be equal for each array sensor); $q_k(f)$ - is a $3m$-dimensional vector frequency response of the medium for k-th source of coherent noise.

If of diffusion noise is absent, i.e. $\sigma^2 = 0$ and a number of coherent noise sources is fewer than that of sensors (in our case, $s < 3m$) the matrix $\boldsymbol{F}(f)$ became singular. It is shown [39,40] that if diffusion noise power tends to zero $(\sigma^2 \to 0)$

$$\phi^*_r(f) = \lim_{\sigma\to 0}\phi_O(\omega) = \left[h^*_w(f,p)\boldsymbol{B}_S(f)\right]\Big/\left[h^*_w(f,p)\boldsymbol{B}_S(f)h_w(f,p)\right], \quad (13)$$

where $\boldsymbol{B}_S = [\boldsymbol{I} - \boldsymbol{R}_S(\boldsymbol{R}^*_S\boldsymbol{R})^{-1}\boldsymbol{R}^*_S]$, $\boldsymbol{I}$ - is a unit matrix.

In the most significant particular case, with only one source for coherent noises $(s=1)$

$$\boldsymbol{B}_1(f) = \left[I - q(f)\boldsymbol{q}^*(f)\Big/q(f)^2\right]. \quad (14)$$

In the absence of diffusion noise, the noise at the output of group filter with frequency response by eq.(11) is equal to zero:

$$\eta(f) = \phi_r(f)\,\boldsymbol{Q}_S(f)\,\zeta(f) = 0, \quad (15)$$

where $\zeta(f) = (\zeta_k(f),\ k=\overline{1,s})$ is a vector of processes in the noise sources. Eq.(13) results from the equation: $\boldsymbol{B}_S(f)\,\boldsymbol{Q}_S(f) = 0$, which is very easy to prove for $s = 1$:

$$\boldsymbol{B}_1(f)\,\boldsymbol{Q}_1(f) = \left[I - q\boldsymbol{q}^*q\Big/|q|^2\right] = 0 \quad (16)$$

Thus, in the absence of diffusion noise, i.e. when seismic noise is fully coherent, they will be suppress by the group filter with frequency response by eq. (13). We call this group filter as the 'spatial rejection group filter'(SRGF).

For suppression of an interfering plain wave arriving from assigned direction the residual beamforming (RB) method is often used [8,23]. Here the beam steered to the "noise direction" is subtracted from every array channel to create residual traces which subsequently are used to form a new 'residual beam' trace. This method can be formulated as the implementation of group filtering procedure by eq.(1) with the vector frequency response

$$\phi_{br}(f) = \boldsymbol{B}_1(f)\,h_w(f,p)\,, \quad (17)$$

where $\boldsymbol{B}_1(f)$ is the matrix by eq.(14) with vector 'phase shifts' $q(f,p)$, corresponding to slowness vector p of an interfering wave. We call this group filter as beam residual group filter (BRGF). Both filters: SRGF and BRGF suppress (theoretically - completely eliminate) the interfering plane wave arriving from the assigned direction. For the BRGF this can be shown by the same way as for the SRGF. But the BRGF method has the disadvantage in comparison with the SRGF that it distorts a frequency content of signal waveform and hence can not be regarded as a 'pure spatial' filtering. For 'pure' plane wave input signals with the correct 'phase shifts' $h_{Wk}(f,p)$ the output of BRGF is not equal to the original waveform:

$$z(t) = h^*_W(f,p)\, \boldsymbol{B}_1(f)\, h_W(f,p)\, s_W(f) \neq s_W(f) \qquad (18)$$

In contrast in this situation, the SRGF produces the undistorted signal waveform. This can be shown in exactly the same way as for the OGF output (see eq. (7)).

Note that coherent noise suppression with the help of the SRGF eq.(13) calls for information on the of frequency response vector $q_k(f)$ of noise propagation paths in the medium. This causes serious problems for its practical implementation.

In some cases coherent noise can be considered as caused by some surface wave (i.e $k=1$). If ray propagation approximation is valid, then in such case the vector $q_1(f))$ can be found from eq.(2.10) where p is an apparent slowness vector of coherent noise wave, $b_i(p,v_W)$ are determined by the type of a noise wave with the help of eq.(2.6b) or (2.6d). The apparent slowness vector p of such interfering noise wave can be estimated with the help of F-K analysis of array records made at a time interval where only coherent noise are present. The another variant of the adaptive SRGF can be obtained if one uses as the of vectors $q_k(f)$ estimates the principal eigen vectors of a MPSD of coherent noise records. The evaluation of the MPSD is preferably to be done by multidimensional ARMA-modelling of the noise time series.

Nevertheless the design of spatial rejection filter using eq.(13), (14) always suffers from some uncertainty with respect to a number of coherent noise waves, their types and parameters p, v_W. The experimental research of such filters as applied to small aperture seismic arrays of NORESS type showed that the zones of noise suppression over the apparent slowness plane (p_x, p_y) are rather, narrow for any SRJF (although are deep enough). Therefore, a SRGF ensures a strong suppression of coherent noise only if medium frequency response vectors $q_k(f)$ used for the noise waves characterisation precisely correspond to reality. In particular, for case of plain interfering waves and the ray propagation approach, one should know accurately the apparent velocities of coherent interfering waves and be sure that the waves are really close to plain. Naturally, all these conditions are seldom met in practice.

At the same time, if one use for coherent noise suppression the adaptive optimal group filter in its canonical version eq.(6), this ones calls for the estimation of matrix power spectral density (MPSD) $\boldsymbol{F(f)}$ of noise at array sensors, irrespectively of a physical essence of the noise origin. It is obvious that in practice there is no ideally coherent noise and $\boldsymbol{F(f)}$ is always non singular (although it may be poorly conditioned). Therefore, in practice the suppression of coherent noise with the help of the AOGF is computationly correct procedure (provided the calculations are accurate enough, that is quite feasible task for present-day computers).

4. Statistically optimal detection of seismic signals

A peculiarity of the seismic signal detection problem is that, while a noise power spectrum can be preliminary estimated so presumed known, a signal spectrum is a priori unknown. The detection procedure currently used for array data processing overcomes this problem by incorporating a set of narrow-band filters followed by the well-known STA/LTA detectors. This procesing is usually applied to a set of beam traces calculated for different steering directions using records of different subarrays. This rather unwieldy array detector demands the adjustment of thresholds and subarray configurations to ensure a good performance [47,48,61].

We propose to implement instead conventional detector an adaptive broad-band statistically optimal detecting procedure following the AOGF noise suppressing procedure as it is described in [36,41,56]. The detector is based on the Bayesian rule for testing a statistical hypotheses about scalar time series z_i, $i=\{t, t+T\}$ being observed in a moving time window. The "simple" hypotheses

H_0 *: observations* z_i *are a "pure" noise with known power spectrum,* is tested against the "complex" hypothesis

H_1 *:* z_i *contain a random signal with unknown power spectrum .*

The detection algorithm should also incorporates an estimation of scalar noise power spectrum density. This can be made, for example, by AR-modelling [10] a "noise part" of data interval being processed. The simple statistically optimal detection procedure which can be used in the on-line array operational mode should perform firstly the noise whitening filtering of data by the recursive filter based on noise AR coefficients. Thus the initial detection problem is reduced to the problem of detecting a random signal with unknown power spectrum on the background of white noise. To construct a simple detection statistic we then assume that if a signal is present the output of whitening filter can be approximated by an AR-model of order p. Under this assumption the hypotheses H_0 and H_1 can be formulated as following:

H_0 : the output of noise whitening filter η_i , i={t, t+T} is the white noise i.e. can be regardedas the AR-time series with parameters $\sigma = 1$, $a = (a_1, ..., a_p) = 0$, and

H_1 : observations η_i , i={t, t+T} are well approximated by some AR -time series with order P and unknown parameters $\sigma \neq 1$, $|a| > 0$.

If the length T of the moving time window is sufficiently large $(T>>p)$ and a signal-to-noise ratio is small, the likelihood ratio for hypothesis H_0 , H_1 can be approximated as follows [36,41]

$$l_t = C \exp\{ b^T D_t - (T/2)| b |^2 \} \qquad (1)$$

where C is a constant which does not depend from observations and AR-parameters, $b^T = (1/\sigma, -a_1/\sigma, ..., a_p/\sigma)$ and D_t is the vector "asymptotically sufficient" statistic:

$$D_t = (D_{t,0}, ..., D_{t,p})^T, \; D_{t,k} = \sum_{i=1}^{T} \eta_{t+i} \eta_{t+i-k} \qquad (2)$$

It is known that if the likelihood ratio for the hypothesis H_0, H_1 has the form eq.(1) there exists this test which has a number of optimal properties for testing the hypotheses [60,36]. The test is

$$\textit{signal is absent if } d_t < K \; ; \quad \textit{signal is present if } d_t > K \qquad (3)$$

where

$$d_t = | D_t |^2 = \sum_{k=0}^{p} D_{t,k}^2 \qquad (4)$$

When a signal is absent, the statistic d_t has approximately the χ^2 probability distribution with $p+1$ degrees of freedom [60]. This gives one the opportunity to calculate the threshold K in eq.(3) providing a given false alarm probability.

The seismic array detector described here provides an alternative to the conventional seismic array detecting procedure. The single adaptive broad-band noise whitening group filter should be used instead of sets of subarray configurations and narrow bands filters commonly being implemented for noise suppression. Then the single broad-band statistically optimal detector should be applied instead of a set of narrow-band STA/LTA detectors with different thresholds. In our opinion, the statistical adaptive strategy formulated ensures the best array signal detection performance where the ambient noise is non-stationary.

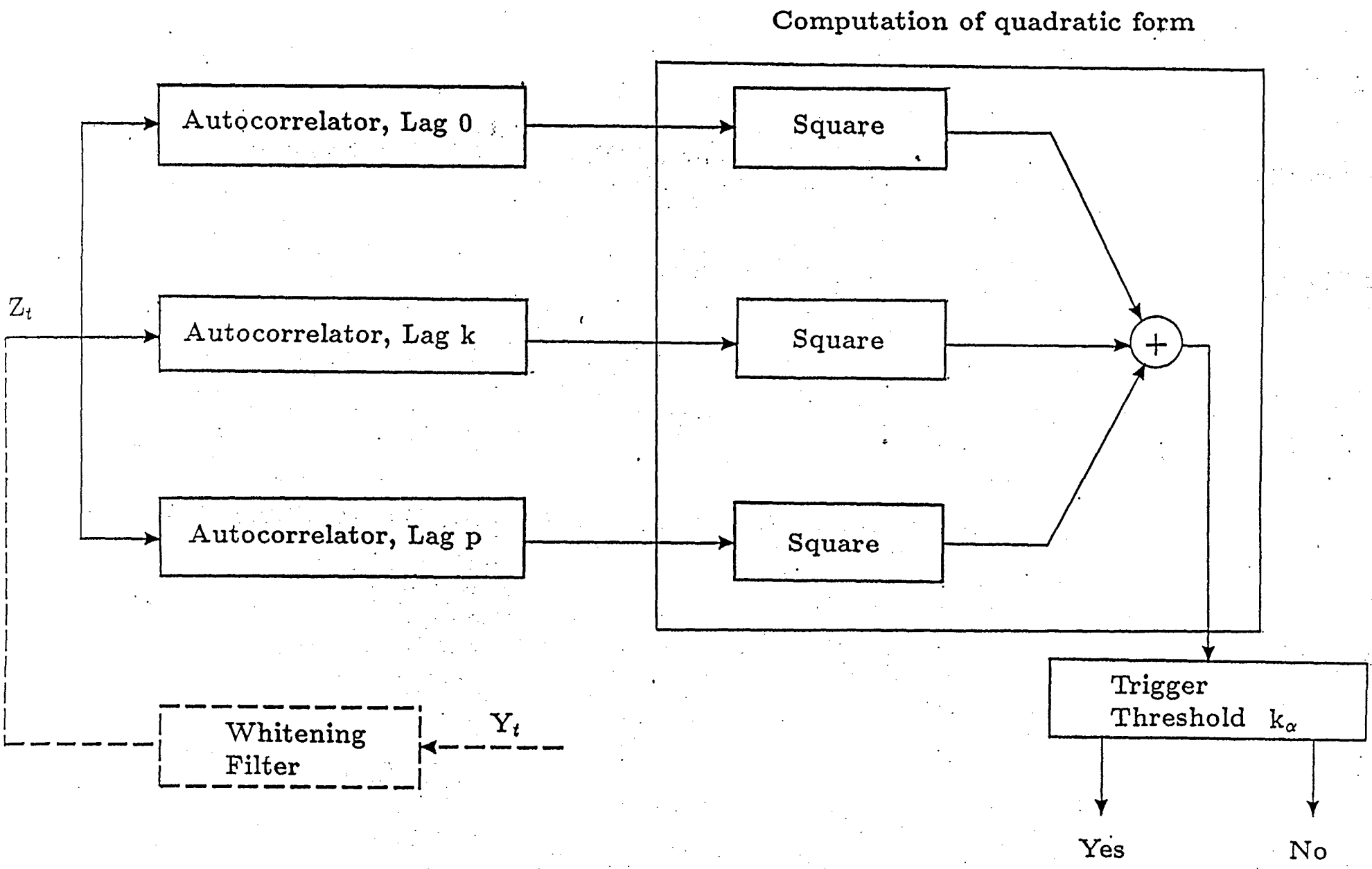
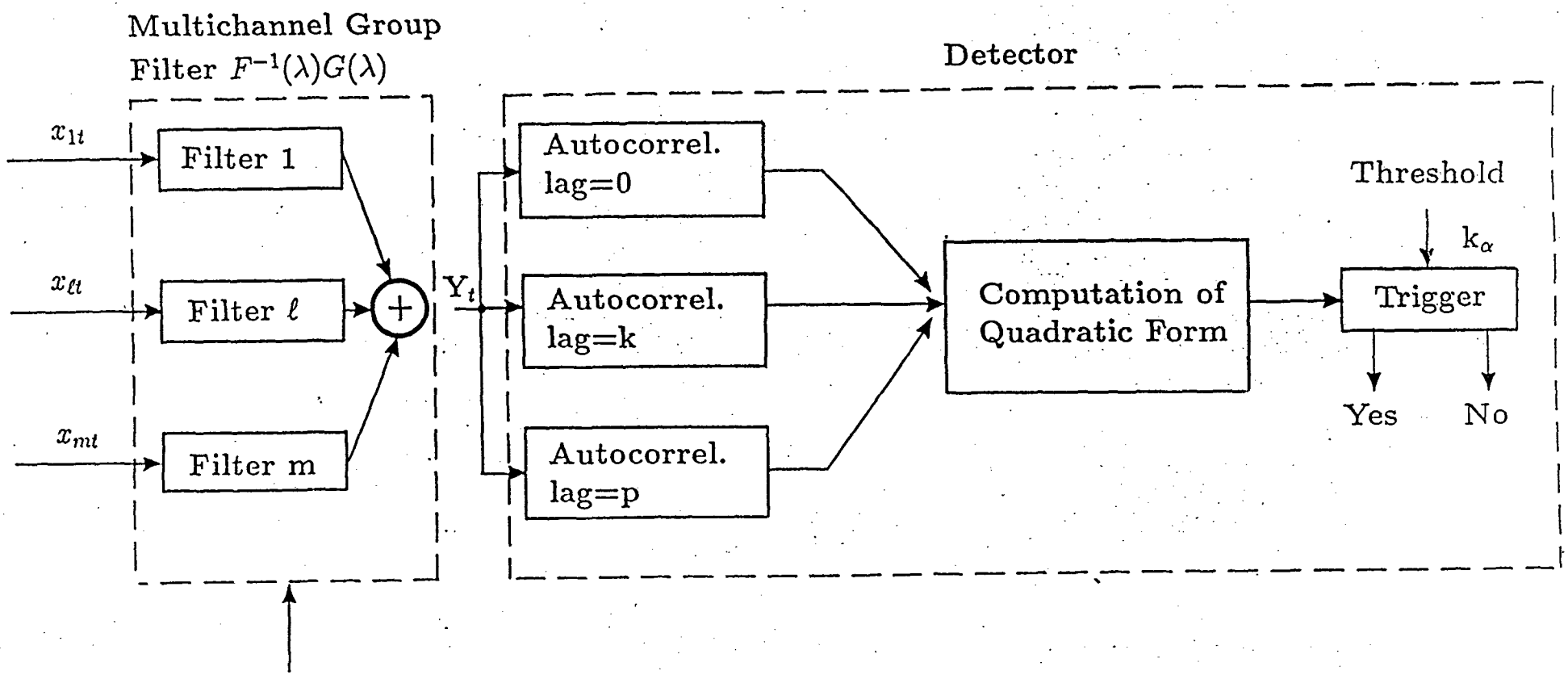

Computation of quadratic form
Autocorrelator, Lag 0
Square
Z_t
Autocorrelator, Lag k
Square
+
Autocorrelator, Lag p
Square
Trigger
Threshold k_α
Whitening
Filter
Y_t
Yes
No

Fig. 4.1.



$$\widehat{F}^{-1}(\lambda) = \sum_{k=-q}^{q} L_k e^{ik\lambda} = (\sum_{1=O}^{q} A_\ell e^{-i\ell\lambda}\, D^{-1} (\sum_{\ell=0}^{q} A_\ell e^{-i\ell\lambda})^*$$

$$G(\lambda) = (exp(-i\tau_1\lambda),\ 1 = \overline{1,m})$$

Fig. 4.2.

5. Statistically optimal estimation of onset times of seismic phases

A seismic source location using records from single array can be accurate enough only if precise estimation is provided for onset times of different seismic phases like P_n, P_g, S_N, L_g and so on. For this purpose we propose to use a statistically optime onset time estimator developed for treating broad-band wavetrains. For array data processing it should be fit on the output of the BF, or AOGF or SRGF noise suppressing procedures. Also this method in its 3-component version can be adopted for processing data from single 3-component seismometer. The algorithm is based on the treatment of signal onset time as a moment when the statistical features of the time series being observed are abruptly changed by a signal arrival. We thus use the following statistical model of observations:

for $t_1 < t < \tau$ *time series* $z_t = n_t$ *is Gaussian noise with a power spectral density* $f_1(f)$;

for $\tau < t < t_2$ *time series* $z_t = n_t + s_t$ *is a signal plus noise Gaussian process with a power spectral density* $f_2(f)$,

where (t_1 , t_2) is the time interval (known from a signal detection procedure) containing a signal onset; τ is an unknown onset time to be estimated. A simple and useful estimator for τ was designed on the basis of autoregressive approximation of the unknown spectral densities $f_1(f)$ and $f_2(f)$. It proved to be effective for seismic phase onset time determination [43,49,50]. This is a maximum likelihood algorithm that calculates the likelihood function $L(\tau)$ of an onset time τ at the interval (t_1 , t_2) and seeks the global maximum of this function to assert the moment τ_{max} as the estimate of an onset time. Calculations are made iteratively for all τ from t_1 to t_2. At each step autoregressive models of observations in intervals (t_1 , τ) and (τ , t_2) are calculated with the help of the Levinson-Durbin procedure [10]. Then $L(\tau)$ is evaluated as [57,41]:

$$L(\tau) = - \tau \ln\sigma_1(\tau) - (n - \tau) \ln\sigma_2(\tau) , \qquad (1)$$

where σ_i^2, $i \in 1,2$ are variances of the autoregressive model residuals, $n = t_2 - t_1$ is a number of samples. The maximum of $L(\tau)$ is found numerically.

A multidimensional version of the algorithm described has also been developed for three component seismogram processing [57]. In this case the onset time is estimated as the moment when the matrix power spectrum of the three component time series being observed is abruptly changed by a signal arrival. The algorithm takes into account changes not only of a power and frequency content, but also of a polarisation of three component observations.

An intensive study of the maximum likelihood algorithm for onset time estimation was recently undertaken in [49,50]. The performance of this algorithm was compared with the manual onset time reading and the conventional SigPro algorithm installed in the Intelligent Monitoring System [3,4]. It was found that in order to make the maximum likelihood onset time estimator the most successful while processing any types of signals one have to implement some preliminary processing:

1. to process the data with a **noise whitening** filter;
2. to apply to output data of the noise whitening filter a **band-pass filter** matched with the signal frequency band;
3. **to decimate** the data in accordance with the highest signal frequency.

6. High resolution F-K analysis procedures for estimation of wave azimuth and apparent slowness

An estimation of azimuths and apparent velocities of several seismic phases from event wavetrains is crucial for performing a location of event sources on the basis of data from single array. In the case of 1-component arrays the estimation is usually made by F-K analysis of array data. The azimuth and apparent velocity of seismic phase are evaluated as the arguments of point of maximum in the F-K spectrum map calculated for given phase records [60,55]. Though the conventional broad-band F-K analysis algorithm proved to be robust and sufficiently accurate for small aperture array data processing [55,6] its application in some specific cases meets some problems. Firstly, it does not provide the sufficient accuracy if a number of array sensors is not large enough (less than ten). Secondly, it sometimes fails to resolve close signal and noise peaks while analysing of a weak seismic wave obscured by a strong coherent interfering wave. Besides it gives in this case the biased estimates of signal arrival parameters. Such the problem arises for example if one needs to detect an explosion hidden in an earthquake wavetrain or a weak seismic phase obscured by a coda of strong previous phase [45].

We propose for such applications to use the modification of high resolution F-K analysis algorithm [18], which utilises the ARMA-estimate of inverse MPSD $\hat{F}^{-1}(f)$ of array data. The algorithm evaluates the F-K map $\hat{P}(f,p)$ as following [38,39]:

$$\hat{P}(f,p) = [\, h^*(f,p)\ \hat{F}^{-1}(f)\ h(f,p)\,]^{-1} \qquad (1)$$

where

$$h(f,p) = (\exp\{i2\pi f u_i^T p\},\ i \in 1,\ldots,m),$$

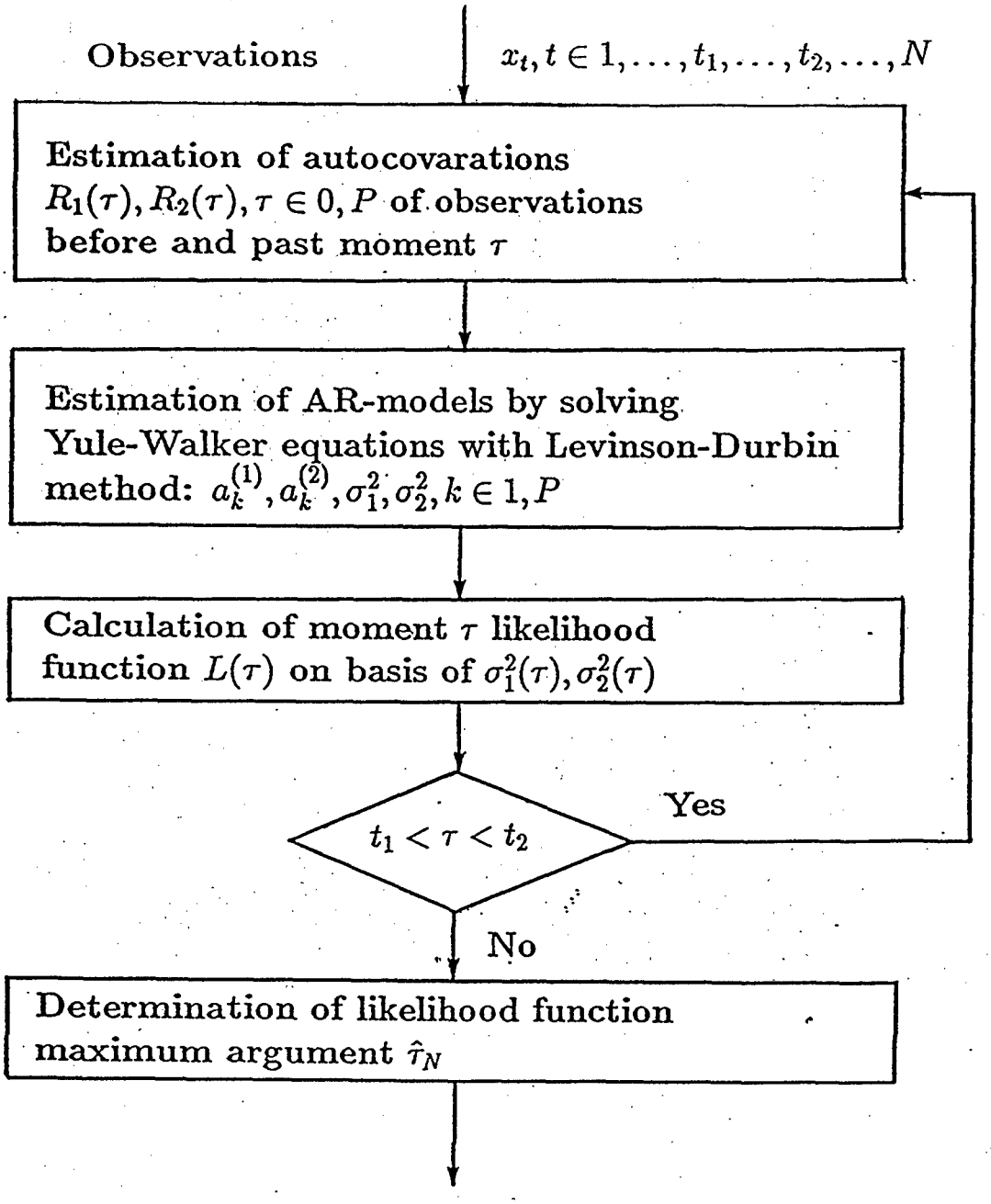

Observations
$x_t, t \in 1, \ldots, t_1, \ldots, t_2, \ldots, N$
Estimation of autocovarations $R_1(\tau), R_2(\tau), \tau \in 0, P$ of observations before and past moment τ
Estimation of AR-models by solving Yule-Walker equations with Levinson-Durbin method: $a_k^{(1)}, a_k^{(2)}, \sigma_1^2, \sigma_2^2, k \in 1, P$
Calculation of moment τ likelihood function $L(\tau)$ on basis of $\sigma_1^2(\tau), \sigma_2^2(\tau)$
$t_1 < \tau < t_2$
Yes
No
Determination of likelihood function maximum argument $\hat{\tau}_N$
$\hat{\tau}_N$ — estimator of signal onset time

Fig. 5.1.

u_i is a vector of coordinates of *i-th* sensor, $p = (p_x, p_y)$ is an apparent slowness vector; $\hat{F}^{-1}(f)$ is the ARMA estimate of inverse MPSD of the phase signals being processed.

Note that $\hat{P}(f, p)$ in eq. (1) is the estimates of F-K spectrum smoothed over some frequency interval $(f- \Delta f, f+\Delta f)$. This is due to the feature of ARMA-estimate of inverse MPSD $\hat{F}^{-1}(f)$ be smoothed over some frequency window. A value of Δf depends on the order of multidimensional ARMA-model. An increase the order decreases Δf . Thus the F-K spectrum estimate eq. (1) is not a narrow-band one. We call it the smoothed high resolution F-K (SHR F-K) estimate.

In some applications there is a possibility to estimate the MPSD of interfering waves using array records at a time interval where a signal is absent. For these applications adaptive statistically optimal estimators of the signal apparent slowness vector p, , can be developed that incorporates the noise MPSD estimate. A detailed discussion of statistical methods for resolving this problem are given in next Section. Here we provide a very brief overview of some results.

From the statistical point of view, the determination of signal apparent slowness vector p is hampered by the lack of information about a signal waveform $s(t)$. The latter can be regarded as a set of "nuisance" parameters involved in the slowness estimation problem. To overcome this obstacle the statistically optimal p-estimator have to incorporate an evaluation of the nuisance parameters [39]. It is noteworthy that the two theoretical assumptions about the signal waveform exist:

1) $s(t)$ is a sample function of a random Gaussian time series with unknown power spectrum,

2) $s(t)$ is a completely unknown deterministic time series,

and both of them lead to the same statistically optimal estimator of an apparent slowness vector. The maximum likelihood approach under the assumption 2) above gives the following result: the ML-estimate of signal slowness vector is to be calculated as the p value maximising the functional

$$L_{ML}(p) = \sum_{f=f_{\min}}^{f_{\max}} \frac{|h(f,p)\hat{F}^{-1}(f)x(f)|^2}{h^*(f,p)\hat{F}^{-1}(f)h(f,p)} \qquad (2)$$

where summation is made over the set of discrete frequencies inside a frequency band chosen; $\hat{F}^{-1}(f)$ is the inverse MPSD of noise. The same p vector estimate one can derived by the maximum likelihood approach using the assumption 1) above with additional constraint that the signal-to-noise ratio is rather large.

If a power of the coherent interfering wave is much greater than the diffusive seismic noise background, so really this wave obscures the signal, we can substitute in eq.(2) instead of $\hat{F}^{-1}(f)$ the estimate of rejection filter matrix $\hat{B}(f)$ (see eq.(3.14)). This matrix is determined by the vector $q(f, p_{int})$ of "phase shifts" for the

interfering wave signals; the apparent slowness p_{int} can be determined with the help of F-K analysis of this signals. Another way for the vector $q(f, p_{int})$ determination is the singular value decomposition of MPSD of the interfering wave records.

Note that the adaptive statistically optimal procedures described are distinct from F-K analysis of "beam residuals" which is often used for the purpose to suppress the strong peak in F-K map caused by interfering wave, and to extract the peak produced by weak seismic phase [8,23]. The conventional F-K analysis of "beam residuals" implies a calculation of the map:

$$R(p) = \sum_{f=f_{min}}^{f_{max}} |h^*(f,p)B(f)x(f)|^2 \qquad (3)$$

The vector p which maximises the functional $R(p)$ is recorded as the estimate of signal wave slowness vector.

The more sophisticated procedure is a high resolution F-K analysis of "beam residuals". This procedure consists of the following steps:

a) calculating a vector time series of "beam residuals" $y(f) = \boldsymbol{B}(f)\, x(f)$;

b) estimating an inverse MPSD $\hat{F}_y^{-1}(f)$ of this time series by multidimensional ARMA-modelling;

c) calculating a HR F-K map via equation (1) by substituting in it the matrix $\hat{F}_y^{-1}(f)$ instead the matrix $\hat{F}^{-1}(f)$;

d) evaluating the vector p maximising the HR F-K map.

Testing the algorithms described revealed that the main disadvantage of the F-K analysis of "beam residuals" is that it gives the biased estimate of signal slowness vector in the case of small and medium SNR, the bias depends on SNR. The statistically optimal algorithm eq.(2) improves the estimation by eliminating the bias and encreasing the resolution power.

Let us consider finally the situation where one suspects that several signal plain waves can simultaneously arrive to the array and be observed on the background of coherent seismic noise. In this case the first problem is to detect each the wave and only the second - to estimate its azimuths and slownesses. This is the more complex problem than that was discussed above. If the signal-to-noise ratio is rather small and powers of different signals are not identical the high resolution F-K analysis can help in this problem mainly to evaluate the number of signal waves and to **preliminary** estimate their azimuths and slownesses. When the amount of signal waves k is known a more precise estimation procedure for estimation of signal azimuths and slownesses can be proposed. The ML approach involving the estimation of 'nuisance' unknown signal waveforms allow us to derive the following functional (we give it below for the case k=2):

$$Q(p_1,p_2) = \sum_{f_j=f_{\min}}^{f_{\max}} \{ x^*(f_j)\, \mathbf{A}(f_j, p_1, p_2)\, x(f_j) \}, \qquad (4)$$

where

$\mathbf{A}(f, p_1, p_2) = \mathbf{F}^{-1}(f)\mathbf{H}(f,p_1,p_2)\, [\mathbf{H}^*(f,p_1,p_2)\mathbf{F}^{-1}(f)\mathbf{H}(f,p_1,p_2)]\, \mathbf{H}^*(f,p_1,p_2)\, \mathbf{F}^{-1}(f);$

$\mathbf{H}(f,p_1, p_2) = [h_{S1}(f, p_1), h_{S2}(f, p_2)]$ is the $m \times 2$-matrix composed by two the column vectors of signal shifts $h_{S1}(f, p_1)$ and $h_{S2}(f, p_2)$, corresponding the azimuths and slownesses of *1-st* and *2-nd* signals and calculated by eq.(1); $F^{-1}(f)$ is the inverse MPSD of array noise, the same as in eq.(2), (3); f_{min}, f_{max} are the margins of signal frequency range, f_j are the Discrete Fourier Transform frequencies. If the signal interfering plane wave with the known azimuth and slowness obscures the signal waves the inverse MPSD $F^{-1}(f)$ can be replaced in eq.(4) by the spatial rejection matrix $B(f)$.

The ML-estimates $\hat{p}_1$ and $\hat{p}_2$ of the azimuths and slownesses of the two signal waves simultaneously arriving to the array are the results of maximising the functional $Q(p_1, p_2)$ over its arguments p_1 and p_2 ; i.e. $(\hat{p}_1, \hat{p}_2)$ is the pair of vectors for which the maximum of $Q(p_1, p_2)$ is attained. Evidently, the calculation of the 'map' of $Q(p_1, p_2)$ -functional depending from four coordinates $(p_{x1}, p_{y1}, p_{x2}, p_{y2})$ is not the adequate procedure in this case. Any conventional recurrent optimisation procedure can be implemented for the evaluation of $\hat{p}_1$ and $\hat{p}_2$ estimates that provide the maximum of functional eq.(4). Note, that functional eq.(4) provides the generalisation of the results of the paper [26] for that case where a coherent component of array noise is strong enough and should be taken into account.

7. Estimation of plane wave apparent velocities based on data from three-component seismic array as statistical problem with nuisance parameters

Introduction.

The experience of implementation of automatic regional seismic monitoring systems revealed that an accuracy of seismic event source location may be significantly enhanced by the usage of information on azimuth and apparent velocity of seismic phase arrivals [16,11]. In principal, should we admit a definite regional model of the Earth, it is possible to locate a sources based on data from single seismic station, if an azimuth and onset times of *P* and *S* phases are measuring [65,66,72]. In such case location errors are depend upon the accuracy of estimation of azimuths and onset times of seismic phases and upon the correctness of

identification of this phases. The most reliable characteristic for seismic phase identification was proved to be the apparent slowness [55]. Hence, estimation of azimuths, apparent slownesses and onset times of seismic phases provides the important information to resolve the main problem of seismic monitoring, namely seismic event location on the basis of seismic observations from a single site.

To estimate a phase arrival direction (AD): azimuth and apparent slowness, one can use the polarisation analysis techniques based on 3-component seismometer records [62,63]. However, an implementation of small aperture seismic arrays results in a significantly higher accuracy of AD estimations [6,71] . As it is demonstrated in Section 2, an apparent slowness of seismic phase can be estimated with the help of array data even in a case the value of wave velocity in the subsurface layer of the Earth is unknown. The advantages of small aperture arrays are essential while monitoring weak seismicity as well as for the purpose of the underground nuclear tests control. For the purpose of monitoring a single small aperture array has proved to be competitive with a local and even regional network of three-component stations [11,55].

In this section the evaluation of azimuth and apparent slowness of a plane wave using data from *3-component* small aperture seismic array is treated as a statistical problem of estimation of multidimensional stochastic time series parameters. This approach is new and distinct from the conventional one, according to which the evaluation of arrival directions is interpreted in the framework of spatial spectral (F-K) analysis [55,60]. As an exception we should note the recent paper [26], where quality of AD estimation based on data from 1-component array and a 3-component seismometer was investigated in regard of reaching statistical bounds of estimation accuracy. However, this paper mainly describes experimental results and does not content a derivation of theoretical formulas for covariance matrices of AD estimates based on array data. Only one estimation algorithm was considered without any discussion of conditions on which its implementation is justified. And though it was mentioned that this algorithm is not optimal, i.e. its accuracy does not reach the statistical bound, no alternative estimates were offered. It is important also that in [26] the significant restrictions on seismic noise were assumed: noise processed were supposed to be white ones and uncorrelated in different array sensors. In practice, however noise often appears to be composed by coherent interfering waves, generated by surf on seashores and/or industrial sources. The another example of strong coherent noise is wave phases of the same or another seismic event interfering with an analysed seismic phase (for instance, such phases as P_n and P_g, having different apparent velocities are interfering at a definite distance from a source). At last, there is a case of special interest where an underground nuclear explosion is performed in a moment of arrival of a seismic phase from a strong earthquake. In such a case body waves from an earthquake interferes the body wave from an explosion but these waves have, in general,

different arrival directions in respect to an array. So an analysis of the problem of plane wave direction estimation in condition of strong coherent noise has the important practical meaning.

7.1 Mathematical models of observations as a random time series with informative and nuisance parameters.

The mathematical model of seismic signals and noise in array sensors was discussed in Section 2. Let us remember that in the time domain this model is

$$x(t) = y(t) + \xi(t) = h(t, p, v_w) * s_w(t) + \xi(t) \qquad (1.1)$$

where * is the sign of convolution of time series, the other notations in eq.(1.1) are the same as in eq. (2.10) and eq. (2.17).

Let us make some remarks concerning a signal *s(t)*, that is a time function (waveform) of seismic phase. They are essential in regard of a problem of seismic wave AD estimation. In this paper we use to make two alternative assumptions about *s(t)*:

a) A signal *s(t)* is a realisation of a Gaussian stationary random time series with zero mean and a power spectral density $v_s(f)$.

Though this assumption may seem to be artificial from the point of view of seismological practice, actually it means that during a synthesis and analysis estimation algorithms we confine ourselves by taking into account only an averaged signal power spectrum without any consideration of signal phase spectrum. As a rule in practice the latter is completely unknown and can not be used. From this point of view the assumption that a waveform *s(t)* is a time function to be realisation of a Gaussian stationary random time series and is a mere a proper way to enable us to use the technique of statistical time series analysis [39,40]

b) *s(t)*is an unknown deterministic time series.

In seismological practice the problem of seismic wave AD estimation has often to be resolved without any knowledge of waveform *s(t)*. The reason is that the seismic event waveforms (and their power spectra) strongly vary for one case or another. Very essential also that a signal *s(t)* is as a rule a broadband one, and its modelling with the conventional Berlage function

$$s(t) = A \exp[-\alpha t] \cos(f_0 t + \varphi) \qquad (1.2)$$

where f_0 represents the central frequency for a wave spectrum, is appeared to be an unjustified simplification (the more so f_0 and α are also unknown a'priori).

With the purpose to estimate the informative parameters $p=(p_x, p_y)$ of observations fitting eq.(1.1) in condition a 'priori uncertainty' of a signal waveform we have to introduce unknown nuisance parameters of the signal. A seismic wave velocity in the surface layer of the Earth crust is often also unknown and in the case

of three-component small aperture arrays one needs to regard it as an additional nuisance parameter.

When modelling $s(t)$ as a random Gaussian time series let us suppose that its power spectral density is a known function $\varphi(f,c)$ depending on q nuisance parameters $c = (c_1,\ldots,c_q)$. The linear model seems to be the simplest in this case:

$$\varphi(f,c) = \sum_{k=0}^{q} c_k \varphi_k(f) \qquad (1.3)$$

where $\varphi_k(f)$ may be typical power spectra of seismic signals for different frequency bands: $\varphi_k(f,c) = \varphi(f - f_k)$, where f_k are central frequencies of such bands. Herewith the fact that $s(t)$ is a broadband signal provides for q in eq.(1.3) to be small, i.e. the parametric representation in eq.(1.3) is efficient with a rather small number q of nuisance.

For the signal model b) where a seismic phase waveform is assumed to be deterministic time series, let us suppose that the all elements of this time series are unknown a'priory. That is in this approach in addition to informative parameters $p=(p_x, p_y)$ we have N unknown nuisance parameters $s(t)$, $t \in 1,\ldots,N$. Thus, for this case the number of nuisance scalar parameters is equal to the number of vector observations $x(t)$, $t \in 1,\ldots,N$.

Thus rigorous statistical approach implies that apparent slowness vector determination using records from a three-component array is the statistical estimation problem involving nuisance parameters. Note that this formulation of the problem is for rather general one because observations $x(t)$, $t \in 1,\ldots,N$ are assumed to be correlated both: different $t \in 1,\ldots,N$ and different coordinates $x_j(t)$, $j \in 1,\ldots,m$ of vectors $x(t.)$ We will suppose below that the matrix power spectral density (MSPD) of noise $\boldsymbol{F}(f)$ is known. Such constraint can be justified in may cases by the ability to observe noise $\xi(t)$ records just before a wave phase arrival. Hence the spectral density of noise may be estimated by means of a special adaptation procedure, discussed in Section 2. The estimate $\hat{F}(f)$ can be used hereafter in algorithms of apparent slowness vector estimation.

The lack of complete information concerning noise power spectral density makes it quite important to investigate the stability of optimal estimation algorithms in regard to deviations of a real noise MPSD from the supposed one. In general this question is discussed in [39], where techniques for such stability evaluation as well as recommendations for design of estimation algorithms, robust in regard to a noise power spectrum are proposed.

Finally, in a case when it seems impossible to estimate a MSPD $\boldsymbol{F}(f)$ using "pure" noise observations, there the is a way to enlarge the space of nuisance parameters of the problem: to introduce a parametric representation of matrix $\boldsymbol{F}(f)$

that involves new nuisance parameters of noise. This approach has been also discussed in [39].

A synthesis of statistically optimal algorithms is based on a criterion of accuracy of estimation which must be properly chosen for a problem under consideration. Hereafter we use as the accuracy criterion for any apparent slowness vector estimate $\hat{p}_N = (\hat{p}_{x,N}, \hat{p}_{y,N})$ the asymptotic covariance matrix of the estimate

$$\lim_{N\to\infty} NE\{(\hat{p}_N - p)(\hat{p}_N - p)^T\} = \Psi_p \qquad (1.4)$$

When N is finite this matrix approximately defines the covariance matrix of an estimate by formula

$$E\{(\hat{p}_N - p)(\hat{p}_N - p)^T\} = \Psi_p / N + \boldsymbol{O}(1 / N^2) \qquad (1.5)$$

where a matrix $\boldsymbol{O}(\alpha)$ is such that $\|\boldsymbol{O}(\alpha)\|/\alpha \to 0$ when $\alpha \to 0$, $\|\boldsymbol{A}\|$ is the Gilbert norm of matrix: $\|\boldsymbol{A}\| = \sum a_{ij}^2$.

The estimate $\hat{p}_N = (\hat{p}_{x,N}, \hat{p}_{y,N})$ of apparent slowness vector p which provides the minimal value of $tr(\Psi_p)$ we will call as the asymptotically efficient (AE) estimate [40]. Note that $tr(\Psi_p)$ is the sum of asymptotic mean square deviations of estimates $\hat{p}_{xN}$, $\hat{p}_{yN}$. from true parameter values p_x , p_y . The criterion of asymptotic accuracy by eq.(1.4) allows one to use analytical techniques of asymptotic estimation theory [5]. This enables one to derive up to the end both synthesis and analysis of the asymptotically optimal estimates and to gain an explicit estimation algorithms and formulas for its asymptotic covariance matrix . At the same time, it is an unresolved task (both from theoretical and practical points of view) to develop an explicit estimation algorithm, which would be the best in terms of any non-asymptotic accuracy criterion, for example, an algorithm having the smallest mean square deviations for any finite sample size N.

7.2 Asymptotically efficient estimates of apparent slowness vector for case random signal waveforms.

Under assumption that the signal $s(t)$ is a stationary zero mean Gaussian random process a vector time series $x(t) = y(t) + h(t,p,v_W)*s(t)$ being observed is a multidimensional zero mean stationary Gaussian time series, which distribution is entirely determined by its matrix power spectral density (MPSD) $\boldsymbol{F}_x(f), f \in [0, f_S/2]$, where f_S is the sampling frequency . With a natural additional assumption that a signal waveform $s(t)$ and noise $\xi(t)$ are statistically independent it is easy to derive that

$$F_X(f) = F(f) + v_S(f) \cdot h(f, p, v_W) \cdot h^*(f, p, v_W) = \\ = F(f) + \varphi_S(f, c) \cdot H(f, p, v_W), \qquad (2.1)$$

where $h_W(f,s,v_W) = (exp[-i2\pi f\,(u_i^T, p_W)]\; b_i(p,v_W),\; i=\overline{1,m})$ is a $3m$-dimensional column vector of medium frequency responses along paths of seismic wave propagation from the first station to the other array stations; $(3mx3m)$ matrix $\boldsymbol{H}(f,p,v_W)$ is equal to : $\boldsymbol{H}(f,p,v_W) = h_W(f,s,v_W)\,[h_W(f,s,v_W)]^*$, $\varphi_S(f,c)$ is a signal power spectral density which has a linear parametric representation in accordance with eq.(1.3). Note, that the MPSD $\boldsymbol{F}_X(f)$ explicitly depends on informative and nuisance parameters of the problem under consideration.

It has been demonstrated in [38,40] that if a probability distribution of observations satisfies rather weak constraints the asymptotically efficient (AE) estimate $\hat{p}_N$ of informative parameter p can be obtained (simultaneously with the estimate $\hat{\theta} = (\hat{c}, \hat{v}_w)$ of the nuisance parameter $\theta=(c,\ v_W)$) by means of the maximum likelihood approach. According to this approach

$$\left(\hat{p}, \hat{\theta}\right) = \arg \max_{p,\theta} \left(L(X_N, p, \theta)\right) \qquad (2.2)$$

where $L(X_N\ ,\ p,\theta) = ln(W(X_N\ ,\ p,\theta))$ is the logarithm of multivariate probability density of observations, $X_N = (x_1^T, \ldots, x_N^T)^T$ is the combined column vector of all observations. The asymptotic covariance matrix of the joint AE estimate $\hat{\vartheta}_N = (\hat{p}_N^T, \hat{\theta}_N^T)^T$ of informative and nuisance parameters is determined by the limit of the normalised Fisher information matrix

$$\Psi_\upsilon^o = \lim_{N\to\infty} N \cdot E_\upsilon\left\{\left(\hat{\vartheta}_N - \vartheta\right)\left(\hat{\vartheta}_N - \vartheta\right)^T\right\} = \Phi_\upsilon^{-1} \qquad (2.3)$$

where $\Phi_\upsilon = \lim_{N\to\infty} (1/N) \cdot \Phi_{\upsilon N}$,

$$\Phi_{\upsilon N} = \left[E_\upsilon\left\{\left(\partial L(X_N, \vartheta)/\partial\vartheta_k\right)\left(\partial L(X_N, \vartheta)/\partial\vartheta_l\right)\right\},\ k,l = \overline{1, q+3}\right] \qquad (2.4)$$

E_υ represents the mathematical expectation with the true parameter values $\upsilon = (p^T, \theta^T)^T$.

For the problems involving nuisance parameters the Fisher matrix $\Phi_{\upsilon\,N}$ and its normalised limit Φ_υ have a block structure

$$\Phi_\upsilon = \begin{bmatrix} \Phi_{pp} & \Phi_{p\theta} \\ \Phi_{\theta p} & \Phi_{\theta\theta} \end{bmatrix} \qquad (2.5)$$

Its inverse matrix has just the same block structure

$$\Phi_{\upsilon}^{-1} = \begin{bmatrix} \Phi_{pp}^{-1} & \Phi_{p\theta}^{-1} \\ \Phi_{\theta p}^{-1} & \Phi_{\theta\theta}^{-1} \end{bmatrix} \tag{2.6}$$

where according to the rules of block-structured matrix inversion [13]

$$\Phi_{pp}^{-1} = \left[\Phi_{pp} - \Phi_{p\theta}^{-1} \cdot \Phi_{\theta\theta}^{-1} \cdot \Phi_{\theta p}^{-1}\right]^{-1} \tag{2.7}$$

It follows from eq.(2.3) and (2.7) that the asymptotic covariance matrix of AE estimate $\hat{p}_N = (\hat{p}_{x,N}, \hat{p}_{y,N})^T$ of informative parameters $p = (p_x, p_y)^T$ can be calculated using the following formula

$$\begin{aligned} \Psi_p^o = \lim_{N\to\infty} N \cdot E_{p,\theta}\left\{(\hat{p}_N - p)(\hat{p}_N - p)^T\right\} = \Phi_{pp}^{-1} = \\ = \left[\Phi_{pp} - \Phi_{p\theta}^{-1} \cdot \Phi_{\theta\theta}^{-1} \cdot \Phi_{\theta p}^{-1}\right]^{-1} \end{aligned} \tag{2.8}$$

From the other hand if there is no nuisance parameters (for example, vector $\theta = (c, v_w)^T$ is known), the asymptotic covariance matrix of informative parameters p is equal to

$$\Psi_p^o = \Phi_{pp}^{-1} \tag{2.9}$$

As $\Phi_{p\theta} = \Phi_{\theta p}$ and the matrix $\Phi_{\theta\theta}$ is positively defined, an important conclusion might be derived:

$$tr\left(\Psi_p\right) = tr\left(\left[\Phi_{pp} - \Phi_{p\theta}^{-1} \cdot \Phi_{\theta\theta}^{-1} \cdot \Phi_{\theta p}^{-1}\right]^{-1}\right) \geq tr\left(\Phi_{pp}^{-1}\right), \tag{2.10}$$

i.e. in a case nuisance parameters are present the sum of asymptotic variances of informative parameter AE estimate is always larger than in the case of lack of nuisance parameters. Eq.(2.8) and (2.9) enable us to calculate the losses in informative parameters estimation accuracy due to a 'priori uncertainty' of the problem compelling us to introduce nuisance parameters.

Note that matrix $\Psi_p(p,\theta)$ in eq.(2.8) determines the lower bound for accuracy of any estimates $\tilde{p}_N(X_N)$ of informative parameters p. It means that for arbitrary estimate $\tilde{p}_N$ and risk function $r[N\, cov_{p\theta}(\tilde{p}_N)]$, where $cov_{p\theta}(\tilde{p}_N) = E_{p\theta}\{(\tilde{p}_N - p)(\tilde{p}_N - p)^T\}$ (the latter must depend on the covariance matrix of estimate and meet some rather weak restrictions [40]), the following inequality is valid

$$\lim_{N\to\infty} r\left[N \cdot \mathrm{cov}_{p\theta}(\tilde{p}_N)\right] \geq r\left[\Psi_p^o(p,\theta)\right], \quad p \in \wp,\ \theta \in \wp \tag{2.11}$$

where $\wp$ and θ are bounded sets in the spaces of informative and nuisance parameters.

Let us construct a computational algorithm for AE estimation of the parameter p, using observations that fit model eq. (1.1). A likelihood function for the observations X_N is the logarithm of Gaussian multivariate probability density of X_N, regarded as the function of parameter p. It can be represented as

$$L(X_N,\upsilon) = -(MN/2)\ln(2\pi) - (1/2)\ln\left(\det\left[C_N(\upsilon)\right]\right) - \\ -(1/2)\cdot X_N^T \cdot C_N^{-1}(\upsilon)\cdot X_N \qquad (2.12)$$

where $\boldsymbol{C}_N$ is a block Teplits matrix of the size mN x mN, composed with the blocks

$$C_\tau = E\left\{x(t)x^T(t+T)\right\} = E\left\{S(t)S^T(t+T)\right\} + E\left\{\xi(t)\xi^T(t+T)\right\} \qquad (2.12')$$

The likelihood function (2.12) depends on the parameters the problems via elements of the inverse matrix $\boldsymbol{C}_N^{-1}(\upsilon)$ and its determinant. It's clear that should the value of mN be of any significance the construction of computational procedures for maximisation of $L(X_N, \upsilon)$ seems to be a difficult task. However, a simple approximation of this functional in the frequency domain is possible if a number of observations N is sufficiently large.

Let us treat the observations which fit the model eq.(1.1) in the discrete frequency domain provided by Discrete Finite Fourier Transform (DFFT) [13]. In this domain the observations are:

$$x_j = y_j + \xi_j \ , \quad j \in \overline{1,N} \ , \qquad (2.13)$$

where N is a number of discrete observations $x_j \Leftrightarrow x(t)$, $y_j \Leftrightarrow y(t)$ and $\xi_j \Leftrightarrow \xi(t)$ are DFFT of $x(t)$, $y(t)$ and $\xi(t)$ $t \in 1,..N$, consequently. DFFT $a_j \Leftrightarrow b(t)$ is determined by the formula

$$a_j = \frac{1}{\sqrt{N}} \sum_{k=1}^{N} b_k e^{ik\lambda_j} \ , \ \lambda_j = 2\pi j/N \qquad (2.13')$$

The well known theorems for Fourier Transform are true for DFFT only in the cyclic form and their conventional expressions are true only asymptotically [13]. For this reason we can write

$$y_j = h_j(p, v_W)s_i + O_j\left(\frac{1}{\sqrt{N}}\right) \qquad (2.14)$$

where $h_j(p,v_W) = (exp(i\lambda_j\, u_k^T p\, f_S)\, b(v_W), \quad k \in 1,...,m), \quad f_S$ is the sampling frequency, $s_j \Leftrightarrow s_W(t)$ is a discrete complex spectrum of a seismic waveform,

$$\max_{j\in 1,N} O_j\left(\frac{1}{\sqrt{N}}\right) \to 0, \quad when\ N\to\infty . \tag{2.14'}$$

The DFFT of array noise ξ_j is a set of normal random vectors with the following characteristics [13].

$$E\{\xi_i\} = 0,\ E\left\{\xi_i \xi_l^*\right\} = \delta_{jl} F_\xi\left(\lambda_j, f_s\right) + O_{jl}\left(\frac{1}{N}\right), \tag{2.15}$$

where $\delta_{jl} = \begin{cases} 1 & j = 1 \\ 0 & j \neq 1 \end{cases}$ is a Kronecker symbol,

$$\max_{l,\ j\in 1,\ N} O_{jl}\left(\frac{1}{N}\right) = O\left(\frac{1}{N}\right). \tag{2.15'}$$

So array noise DFFT frequency elements have weak correlations one with another for large N even if noise observations themselves are significantly correlated in time. I.e. DFFT appears to be an asymptotically decorrelating transform.

For a random seismic waveform eq. (2.13)-(2.15) implies a conclusion that x_j are asymptotically uncorrelated normal vectors with moments

$$E\left\{x_j\right\} = 0,\ E\left\{x_j x_l^*\right\} = \delta_{jl} F_{xj} + O_{jl}\left(\frac{1}{N}\right), \tag{2.16}$$

where $F_{xj} = F_x\left(\lambda_j, f_s, p, \theta\right) = F\left(\lambda_j, f_s\right) + H\left(\lambda_j, f_s, p, v_w\right)\varphi\left(\lambda_j, f_s, c\right)$, are expressed by eq. (2.1) and eq.(1.3).

If to ignore a weak statistic dependence of x_j for different j (i.e. to drop terms $O_{jl}(1/N)$ in eq. (2.16)) one can get the following approximate expression for the likelihood function eq.(2.12) in the discrete frequency domain:

$$\begin{aligned} L(X_N, \vartheta) = \ln W(x_1, \ldots, x_N; \vartheta) \cong -\frac{MN}{2} \ln 2\pi - \\ -\frac{1}{2}\sum_{j=1}^{N} \ln \det F_{xj} - \frac{1}{2}\sum_{j=1}^{N} x_j^* F_{xj}^{-1} x_j \quad , \end{aligned} \tag{2.17}$$

where $\boldsymbol{F}^{-1}{}_{xj} = \boldsymbol{F}^{-1}{}_x(\lambda_j, f_s)$

Sufficiently accurate assessments of residual terms $O_{jl}(1/N)$ in eq. (2.16) enable one to gain the following relationship between the likelihood function by eq.(2.12) in the time domain and likelihood the function by eq.(2.17) in frequency domain :

$$L(X_N, p, \theta) = L_f(X_N, p, \theta) + O\left(\sqrt{N}\right) \tag{2.18}$$

I.e. the "distance" between L and L_f is increasing with an increase of number of samples N. However as the order of magnitude of L_f is $O(N)$ we may consider that L_f contains the "main part " of the likelihood function for the problem under consideration [39].

The following practically important conclusion can be derived from the discussion above. Namely, to obtain the AE estimate of the joint vector parameter $\vartheta=(p, \theta)$ it is enough to get the maximum of approximate likelihood function in frequency domain [39]:

$$\hat{\vartheta}_N = \arg\max_{\vartheta} L_f(x_N, \vartheta) \qquad (2.19)$$

I.e. the algorithm of AE estimation can be constructed as the procedure of maximisation of following functional

$$\Lambda(X_N, \vartheta) = \sum_{j=1}^{N} \ln\det F_{xj}^{-1}(\vartheta) - \sum_{j=1}^{N} x_j^* F_{xj}^{-1}(\vartheta) x_j, \qquad (2.20)$$

where $F_{xj}^{-1}(\vartheta) = Q(p,\theta)F_j^{-1}$; $Q(p,\theta) = \left[I - \dfrac{F_j^{-1} H_j(p, v_w)}{\varphi_j^{-1}(c) + trF_j^{-1} H_j(p, v_w)} \right]$,

subscript j means the dependence of corresponding functions on λ_j. The latter equation in eq.(2.20) is nothing else but the well-known Bartlet formula for inversion of matrices like by eq.(2.1) [13].

Eq.(2.20) for the functional Λ may be transformed to a form which simplifies calculations during an iterative maximisation procedure and explains a "physical" meaning of the functional:

$$\Lambda(X_N, \vartheta) = C + \sum_{j=1}^{N} \ln\det Q_j(p,\theta) - \sum_{j=1}^{N} |z_j(p,\theta)|^2 \qquad (2.21)$$

where $C = \sum_{j=1}^{N} \ln\det F_j^{-1} + \sum_{j=1}^{N} x_j^* F_j^{-1} x_j$ is the term independent on the parameters υ;

$$z_j(p,\theta) = \frac{h_j^*(p, v_w) F_j^{-1} x_j}{\sqrt{\varphi_j(c) + h_j^*(p, v_w) F_j^{-1} h_j(p, v_w)}}$$

is the output signal of conditional optimal Winner group filter. The latter transforms a multichannel input signal to a scalar trace and maximises the signal/noise ratio along this trace under the condition of whitening output noise. Designating

$$T_j = F_j^{-1} x_j x_j^*;\ \Psi_j(\vartheta) = F_j^{-1} H_j(\vartheta);\ \psi_j(\vartheta) = \varphi_j^{-1}(c) + tr\Psi_j(\vartheta), \quad (2.21')$$

we get the formula for the functional Λ which is convenient for calculations:

$$\Lambda(X_N, \vartheta) = C + \sum_{j=1}^{N} \ln\det\left[I - \frac{\Psi_j(\vartheta)}{\psi_j(\vartheta)} \right] - \sum_{j=1}^{N} \frac{tr_j T_j \Psi(\vartheta)}{\psi_j(\vartheta)}. \quad (2.22)$$

During iterative maximisation of functional Λ the matrices T_j (which depend on observations) remain unchanged, the process involves recalculations only of scalars $\psi_j(\upsilon)$ and matrices $\Psi_j(\upsilon)$. A computation of the functional eq.(2.22) maximum is facilitated due to existence of simple explicit formulas for its partial derivatives by parameters p, c and v_w.

It is rather easy to derive an analytical expression for the limit of normalised Fisher matrix for parameters υ [39]. It enable us to calculate (based on eq.(2.8)) the asymptotic covariance matrix for AE estimate of apparent slowness vector

The analysis of eq.(2.21) reveals that in a case where signal/noise ratio is large values $\varphi_j^{-1}(c)$ become negligible for all j in comparison with $h^*_j(p,v_w)\, \boldsymbol{F}_j^{-1}\, h_j(p,v_w)$. Hence, the dependence of functional $\Lambda(X_N, p, c, v_w)$ upon an unknown signal spectrum $\varphi_j(c)$ is weakening up to completely vanish when $\varphi^{-1} \to 0$. (It is not entirely obvious and demands a thorough investigation due to the matrix $\boldsymbol{Q}(p, c, v_w)$ becomes singular for this case. This causes the first term of eq.(2.21) to rise unrestrictedly). So if a signal/noise ratio is large enough the apparent velocity vector AE estimate slightly differs from the estimate minimising the following functional

$$\tilde{\Lambda}(X_N, p) = \sum_{j=1}^{N} \left| \tilde{z}_j(p) \right|^2 \quad (2.23)$$

where $$\tilde{z}_j(p) = \frac{h_j^*(p) F_{\xi j}^{-1} x_j}{\sqrt{h_j(p) F_{\xi j}^{-1} h_j(p)}}$$

where $F_{\xi j} = F(\lambda_j, f_s)$ are values of noise MPSD for DFFT frequencies.

In other words if a signal/noise ratio is large enough one must take into account only the noise matrix spectral density meanwhile an information on signal spectrum become insignificant.

Finally let us consider the functional eq.(2.22) in the particular case when both signal and noise can be assumed to be white random processes and noise vector components $\xi_k(t), k \in 1, \ldots, m$ are mutually uncorrelated i.e.: $F_{\xi j} \equiv I\sigma^2$, $\varphi_j(c) \equiv \sigma_s^2$. In this case

$$\Lambda\left(X_N, p, v, \sigma_s^2\right) = C + \sum_{j=1}^{N} \ln \det \sigma^2 \left[\frac{H_j(p, v_w)}{m + \sigma^2/\sigma_s^2}\right] - \\ -\left[\left(m + \sigma^2/\sigma_s^2\right)\right]^{-1} \sum_{j=1}^{N} \left|h_j(p, v_w) x_j\right|^2 \tag{2.24}$$

For a large signal/noises ratio $\sigma^2/\sigma^2_s << m$ and the AE estimate of vector p can be obtained by minimisation of the functional

$$\tilde{\Lambda}(X_N, p) = \sum_{j=1}^{N} \left|h_j(p) x_j\right|^2 \tag{2.25}$$

So it coincides with the well known apparent velocities estimate gained by the broadband F-K analysis technique [55,71].

In conclusion of this Section let's consider the mathematical expectation of functional $\Lambda(X_N, \upsilon)$ for the case where observations have the fixed parameter value $\upsilon = \upsilon_0$ and (regarded as a function of unknown parameters υ) it can be shown that this function is equal to

$$U(\vartheta, \vartheta_0) = E_{\vartheta_0}\{\Lambda(X_N, \vartheta)\} = C + \beta(\vartheta) - \\ -\sum_{j=1}^{N} \frac{a_j(\vartheta_0) + a_j(\vartheta) a_j(\vartheta_0) - b_j(\vartheta, \vartheta_0)}{1 + a_j(\vartheta)} \tag{2.26}$$

where it is designated

$$a_j(\vartheta) = h_j^*(p, v_W) F_j^{-1} h_j(p, v_W) \varphi(c)$$

$$b_j(\vartheta, \vartheta_0) = h_j^*(p, v_W) F_j^{-1} h_j(p_0, v_W) \left(\varphi(c)\varphi_0(c)\right)^{1/2}$$

$$\beta(\vartheta) = -\sum_{j=1}^{N} \ln \det \left[\frac{F_{\xi j} H_j^{-1}(p, v_w) \varphi(c)}{\varphi^{-1}(c) + tr F_j^{-1} H_j(p, v_w)}\right]$$

A localisation in the parameter space of a maximum of function $U(\upsilon, \upsilon_0)$ over υ and a "width" of this function serves as characteristics quality of AE estimate for a case of finite sample size N. This enables us, for example, to analyse the dependence of apparent velocity estimate accuracy on an array geometry by a rather simple numerical techniques. Such functions are widely used in the radar and sonar applications and are called as the "uncertainty functions" of parameter estimation methods.

7.3 Asymptotically efficient estimates of apparent slowness vector for the case of small signal to noise ratio

The estimates of apparent slowness vector, considered in Section 3.2 are the best in terms of asymptotic quality criterion by eq.(1.4), (1.5). However the usage of numerical technique for finding a maximum of the functional eq.(2.22) makes the computational algorithms rather labour consuming. The iterative estimation procedure is aggravated and slowed down by the necessity to maximise the functional eq.(2.22) in *(q+3)* parameters, *q+1* of which are nuisance, e.g. actually unnecessary for the main estimation problem.

The additional assumption that signal/noise ratio is small allows for a significant simplification of asymptotically efficient estimation algorithm. Under this assumption a mathematically correct formulation of the apparent slowness estimation problem is to analyse the following model of observations

$$x(t) = 1/\sqrt[4]{N}\, s(t) + \xi(t);\ x(t) = \left(x_1(t),\ldots,x_m(t)\right),\ \ t \in 1,\ldots,N \quad ,(3.1)$$

In the framework of this model a matrix power spectral density (MPSD) of the time series $x(t),\ t \in 1,\ldots,N$ is equal to

$$F_x(f) = F(f) + H(f,p,v_w)\sum_{l=1}^{q}\frac{c_l}{\sqrt{N}}\varphi_l(f)\ . \qquad (3.2)$$

It can be shown [39,40] that under weak restrictions on noise MPSD $\boldsymbol{F(f)}$ the likelihood function (LF) eq.(2.12) of observations fitted eq.(3.1), (3.2) have the following asymptotic representation:

$$\begin{aligned} L\left(X_N \middle| c/\sqrt{N}, p, v\right) &= -L(X_N,0) + c^T\delta(X_N,p,v) - \\ &-\frac{1}{2}c^T\Gamma(p,v)c + \alpha_N(X_N,p,v,c) \end{aligned} \qquad (3.3)$$

where

$$\delta(X_N,p,v) = \frac{1}{\sqrt{N}}\sum_{j=1}^{N}\left[\left|h_j^*(p,v)F_j^{-1}x_j\right|^2 - h_j^*(p,v)F_j^{-1}h_j(p,v)\right]\varphi_j$$

is the vector asymptotically sufficient statistic for the small nuisance parameters $c/\sqrt{N}$ [38-40];

$$\Gamma(p,v_w) = \frac{1}{N}\sum_{j=1}^{N}\left[h_j^*(p,v)F_j^{-1}h_j(p,v)\right]^2\varphi_j\varphi_j^T$$

is the limit of normalised Fisher matrix for the small nuisance parameters $c/\sqrt{N}$;

$\alpha(X_N, p, c, v)$ is a residual term of the LF asymptotic decomposition, converging to zero in probability as a random process in the space $\boldsymbol{C}[\wp \times \theta]$ of continues functions from (p, θ) with a uniform metric, where θ is a bounded set of nuisance parameters $\theta = (c, v)$, $\wp$ is a bounded set of informative parameters p ; $L(X_N, 0)$ is the likelihood function of "pure" noise; due to independence of this term on the parameters p, c, v it may be dropped hereinafter.

As follows from eq.(3.3) the AE parameter estimate for observations fitted the model eq. (3.1), (3.2) has the form

$$\tilde{\vartheta}_N = \left(\tilde{\theta}_N, \tilde{p}_N\right) = \arg\max_{c,v,p}\left[c^T \delta(X_N, p, v) - \frac{1}{2} c^T \Gamma(p, v) c\right] \quad (3.4)$$

Note that asymptotic factorisation by eq.(3.3) can be obtained not rigorously, if to implement the second-order Teilor decomposition of the $L_f(X_N, p, c, v)$ (asymptotic representation eq.(2.2) of the likelihood function in the frequency domain) at the point $c = 0$ and to substitute the second summand with its mathematical expectation.

The estimate $\tilde{\vartheta}_N$ is significantly simpler than the common case AE estimate by eq.(2.19)-(2.22). Indeed by maximising eq.(3.4) in c with the fixed p and v it is easy to obtain

$$\begin{aligned} \tilde{c}(p, v) &= \arg\max_{c}\left[c^T \delta(X_N, p, v) - \frac{1}{2} c^T \Gamma(p, v) c\right] = \\ &= \Gamma_N^{-1}(p, v)\delta(X_N, p, v) \end{aligned} \quad (3.5)$$

The substitution of eq.(3.5) into eq.(3.4) gives

$$\left(\tilde{p}_N, \tilde{v}_N\right) = \arg\max_{p,v} R(X_N, p, v), \quad (3.6)$$

where

$$R(X_N, p, v) = \delta^T(X_N, p, v)\Gamma_N^{-1}(p, v)\delta(X_N, p, v) \quad (3.7)$$

Thus when a signal/noise ratio is small the problem of apparent slowness vector and seismic wave phase velocity estimation can be reduced to maximisation of functional eq.(3.7) in p and v. This is essentially simpler task than the maximisation of functional eq.(2.22). It is attributed both to the less number of calculations for evaluating the values of $R(X_N, p, v)$ as well as its derivatives by p_x , p_y and v , and to the fact that the procedure of optimisation involves only three parameters: p_x , p_y and v, instead of $(q+3)$ parameters (c, p, v). As a rule this yields to a significant increase in the speed of iterative procedures of numerical optimisation. The maximisation of functional eq.(3.7) becomes easier also due to the existence of simple explicit expressions for its partial derivatives of the first and the second order by parameters.

With the purpose to investigate an asymptotic quality of estimates by eq.(3.6) one needs first to proof the √N-consistency of estimates $(\tilde{p}_N, \tilde{v}_N)$, e.g. the convergence $(\tilde{p}_N, \tilde{v}_N)$ in probability with speed $1/\sqrt{N}$ to (p_0, v_0) that is to the true values of apparent slowness vector and seismic wave phase velocity. Then one needs to find an asymptotic covariance of apparent slowness vector estimate $\tilde{p}_N$ i.e.. the limit

$$\lim_{N\to\infty} E_{\vartheta_0} N(\tilde{p}_N - p_0)(\tilde{p}_N - p_0)^T = K_{\tilde{p}}(\vartheta_0) \qquad (3.8)$$

Comparison of $\mathrm{tr}K(p_0, v_0, c_0)$ with the right part of eq.(2.10) i.e. with the lower bound for estimation errors enables one to determine a possible loss in asymptotic quality of apparent slowness vector estimate by eq.(3.6) comparing with the asymptotically efficient algorithm by eq.(2.19)-(2.22). Such investigations may be performed based upon techniques developed in [39].

7.4 Apparent slowness estimates in case of completely unknown signal waveforms.

If the waveform $s(t)$ of seismic phase is completely unknown (ref. Section 7.1, model b) the spectral samples s_j, $j \in 1,\ldots,N$ in eq.(2.14) are completely unknown too and must be considered as set of nuisance parameters of the problem. If to drop small terms $O_j(1/\sqrt{N})$ in eq.(2.13)-(2.14) then observations in the frequency do main would fit an non-linear regression model with unknown "regressors" s_j. As a number of nuisance parameters s_j tends to infinity with an increase of the number of data samples N, a reasonable question arises: do such formulation of the problem provides existents of any $\sqrt{N}$ -consistent estimate, for which estimation errors tend to zero with N tends to infinity and the asymptotic covariance matrix eq.(1.4) exists. Hereafter we are to construct an example of such estimate by means of the maximum likelihood techniques. The proof of the $\sqrt{N}$ -consistency of this estimate and analytical expression for its asymptotic covariance matrix may be obtained using the results of [39].

As it follows from the example mentioned a class K of $\sqrt{N}$ -consistent estimates exists in the problem under consideration. The another important theoretical question is how to construct a reachable lower risk boundary for estimates from class K (similar to eq.(2.11) boundary). It is clear that for estimation problem being discussed there must exist another expression for the right part of eq.(2.11), distinct from the risk $r\left(\Psi^o(p,v)\right)$. Really, the inequality (2.11) was derived under suggestion that a number of nuisance parameters of the task is definite, and there exists the limit of the normalised Fisher information matrix for the total set of parameters to be estimated.

The theoretical questions discussed above have been investigated in [39] for the general problem of parametric identification of multidimensional linear systems based on observations of system input and output signals distorted by noise. The considered here problem of estimation of seismic wave arrival direction based on array data represents a particular case of mentioned general problem.

Let us derive a maximum likelihood estimate of apparent slowness vector p and wave velocity v for the case of completely unknown waveform s_j $j = 1$,..., N. We will use an approximate expression for the likelihood function in the frequency domain, similar to eq.(2.17). As s_j are deterministic (but unknown) complex values, and ξ_j are Gaussian vectors with moments by eq.(2.16) then the following asymptotic expression for the likelihood function is valid

$$L\left(X_N \middle| p, v, \{s\}\right) = C - (1/2)\sum_{j=1}^{N} \ln \det F_j -$$

$$-(1/2)\sum_{j=1}^{N}\left(x_j - h_j(p,v)s_j\right)^* F_j^{-1}\left(x_j - h_j(p,v)s_j\right) + \mathrm{O}_j\left(1/\sqrt{N}\right) = \quad (4.1)$$

$$= l_f\left(X_N \middle| p, v, \{s\}\right) + \mathrm{O}_j\left(1/\sqrt{N}\right)$$

The maximum likelihood (ML) estimates of informative parameters p, v and unknown nuisance parameters s_j , $j \in 1,...,N$ is the solution of the following set of equations

$$\frac{\partial}{\partial \mathrm{Re}\, s_j} l_f\left(X_N \middle| p, v, \{s_j\}\right) = 0; \quad \frac{\partial}{\partial \mathrm{Im}\, s_j} l_f\left(X_N \middle| p, v, \{s_j\}\right) = 0;$$

$$\frac{\partial}{\partial p_\alpha} l\left(X_N \middle| p, v, \{s_j\}\right) = 0; \quad \frac{\partial}{\partial v} l\left(X_N \middle| p, v, \{s_j\}\right) = 0, \quad (4.2)$$

$$j = \overline{1,N} \ , \quad \alpha \in \overline{x,y}$$

where Res_j and Ims_j are the real and imaginary parts of complex signal spectral samples.

Having positively defined Hermitian matrix $\boldsymbol{F}^{-1}{}_j$ expressed in the form: $\boldsymbol{F}^{-1}{}_j = \boldsymbol{F}^{-1/2}{}_j \boldsymbol{F}^{-1/2}{}_j$ one can write the main term of eq.(4.1) like

$$L\left(X_N \middle| p, v, \{s_j\}\right) = C -$$

$$-(1/2)\sum_{j=1}^{N} \ln \det F_j - (1/2)\sum_{j=1}^{N}\left|n_j - d_j(p,v)s_j\right|^2 \quad (4.3)$$

where

$$n_j = F_j^{-1/2} x_j; \qquad d_j(p,v) = F_j^{-1/2} h_j(p,v) \tag{4.4}$$

The first subsystem of equations (4.2) is linear and as it is easy to verify has the following analytical solution

$$\overset{+}{s}_j(p,v) = \frac{d_j^*(p,v)x_j}{\left|d_j(p,v)\right|^2}, \quad j \in \overline{1,N} \tag{4.5}$$

Substituting $\overset{+}{s}_j(p,v)$ into the second subsystem of eq.(4.2) one obtain that velocity in the problem being considered the ML estimates of apparent slowness vector and wave are the solutions of the following set of non-linear equations

$$\begin{aligned} \rho_\alpha(X_N,p) &= \frac{\partial}{\partial p_\alpha}\left(\sum_{j=1}^{N}\left|\Pi_j(p,v)n_j\right|^2\right) = 0\ , \quad \alpha \in (x,y) \\ \rho_v(X_N,p) &= \frac{\partial}{\partial v}\left(\sum_{j=1}^{N}\left|\Pi_j(p,v)n_j\right|^2\right) = 0 \end{aligned} \tag{4.6}$$

where $\Pi(p,v) = I - d_j(p,v)\left|d_j(p,v)\right|^{-2} d_j^*(p,v)$.

We obtain then the final equations for the ML-estimates of apparent slowness vector and wave velocity by substituting eq.(4.4) into eq.(4.6) and accounting the easy-to-check relationship $\Pi_j \Pi^*_j = \Pi_j$

$$\begin{aligned} \rho_\alpha(X_N,p) &= \sum_{j=1}^{N} x_j \dot{A}_{\alpha j}(p) x_j = 0\ , \quad \alpha \in \overline{x,y} \\ \rho_v(X_N,p) &= \sum_{j=1}^{N} x_j \dot{A}_{vj}(p) x_j = 0 \end{aligned} \tag{4.7}$$

where $\dot{A}_{\alpha j}(p,v) = \frac{\partial}{\partial p_\alpha} A_j(p,v)$; $\dot{A}_{vj}(p,v) = \frac{\partial}{\partial v} A_j(p,v)$,

$$A_{\alpha j}(p,v) = \left(I - \frac{F_j^{-1}H_j(p,v)}{trF_j^{-1}H_j(p,v)}\right)F_j^{-1}\ .$$

Performing differentiation of the matrix $A_j(p,v)$ by p_x, p_y we obtain the following calculation formulas for partial derivatives:

$$\dot{A}_{\alpha j}(p) = \frac{F_j^{-1}}{trF_j^{-1}H_j(p,v)}\left(\dot{H}_{\alpha j}(p) - H_j(p)\frac{trF_j^{-1}\dot{H}_{\alpha j}(p)}{trF_j^{-1}H_j(p)}\right) \tag{4.8}$$

where $\dot{H}_{\alpha j}(p) = \dfrac{\partial}{\partial p_\alpha} H_j(p)$

The all discussed above provides the important conclusion that ML-estimates of seismic wave apparent slowness vector p and velocity v in the case of completely unknown waveform must minimise the functional

$$\Gamma(X_N, p, v) = \sum_{j=1}^{N} x_j^* \mathrm{A}_j(p, v) x_j \qquad (4.9)$$

The functional eq.(4.9) has a clear geometrical interpretation. According to eq.(2.14) the signal frequency sample s_j belong (with accuracy up to $O_j(1/\sqrt{N})$) to some 2 -dimensional subspaces of the complex m-dimensional x_j -vector spaces $\boldsymbol{C}_j^m$. This subspace (depending on j) is determined by the relationship

$$L_j(p, v) = \left\{ h_j(p, v) n_j,\ n_j \in C_j^m \right\} \qquad (4.10)$$

It is easy to demonstrate that the functional eq.(4.9) is the sum of squares of distances from observation frequency samples x_j to the corresponding subspaces $\boldsymbol{L}(p)$ the distances are calculated in the metrics determined by the inner products

$$(a,b) = a^* \boldsymbol{F}_j b,\ a,b \in \boldsymbol{C}_j^m \} \qquad (4.11)$$

Thus in the problem being considered the maximum likelihood techniques brings to estimate of vector (p,v), which is a generalisation of the known estimate by orthogonal statistical regression method.

By a simple transforming of the functional eq.(4.9) one can verify that the method described really provides exactly the same estimates of an apparent slowness vector and velocity as the functional eq.(3.43), which provides the AE estimates in the case of random Gaussian signal with large signal/noise ratio. It is evident that eq.(4.7) and eq.(4.9) result in

$$\Gamma(X_N, p, v) = \sum_{j=1}^{N} x_j^* F_j^{-1} x_j - \sum_{j=1}^{N} \frac{\left| h_j^*(p, v) F_j^{-1} x_j \right|^2}{h_j^*(p, v) F_j^{-1} h_j(p, v)} \qquad (4.12)$$

The first term in eq.(4.12) does not depend on parameters (p, v), so the estimation of p, v is reduced to minimisation of the second term, which coincides exactly with the functional $\widetilde{\Lambda}(X_N, p, v)$ by eq.(2.45). This is the very important coincidence which exhibits the close connection between so different at first glance mathematical models of observations: the model of random Gaussian signal with an unknown power spectrum density and the model of deterministic signal with

completely unknown waveform. This interesting (but natural) correlation of these models has been discussed in [39].

Eq.(4.12) is the helpful for finding of the uncertainty function of ML-estimate. According to the definition stated in Section 3.3 this function is equal to

$$U(p,v,\{s_j\}) = E_{p,v,\{s_j\}}\{\Gamma(X_N,p,v)\} = \\ = \sum_{j=1}^{N} trA_j(p) E_{p,v,\{s_j\}}\{x_j x_j^*\} \tag{4.13}$$

where $E_{p,v,\{s_j\}}$ is the mathematical expectation corresponding the probability distribution of the observations defined by eq.(1.1),(1.2); $\{s_j\}$ is a set of signal spectral samples. According eq.(1.2) we have

$$E_{p,v,\{s_j\}}\{x_j x_j^*\} = F_j + H_j(p,v)|s_j|^2 \tag{4.14}$$

Hence, the uncertainty function is equal to

$$U(p,v,\{s_j\}) = \sum_{j=1}^{N}[trA_j(p,v)F_j + trA_j(p,v)H_j(p,v)s_j] \tag{4.15}$$

The formula eq.(4.15) enables one to estimate qualitatively the accuracy and resolving power of slowness vector and velocity maximum likelihood estimates derived from three component array data.

8. Testing adaptive statistical algorithms using records from Scandinavian seismic arrays

The adaptive optimal group filtering (AOGF), adaptive statistically optimal detection (ASOD), maximum likelihood onset time estimation (MLOE) and smoothed high resolution F-K analysis algorithms were tested using data recorded at the Scandinavian small aperture seismic arrays NORSAR, NORESS, ARCESS and FINESA. The high coherency of noise field in these arrays regions provides a large gain of AOGF over the conventional beamforming procedure (BF), especially if the processing of these array data is accomplished in the broad frequency band. Numerous experiments were carried out to test the AOGF capability to suppress seismic noise recorded at NORESS, ARCESS and FINESA in various time of day and in different seasons. Some examples of results of these experimental are presented below.

Fig.8.1 shows the result of processing by the AOGF and BF procedures of a 40 min. NORESS "pure" noise record made in April 1990 (the frequency band 0.2-5

Hz). The AOGF adaptation has been done using the first 2 minutes of record. The evident stability of the noise suppression by the AOGF proves a stability of noise statistical characteristics on an our basics and allow one to hope that applications of the AOGF in on-line detection procedures would not demand very frequent re-adaptation of the group filter. This is practically important because the AOGF adaptation is a time consuming procedure.

The experiments made for extracting seismic phases of local and regional events from array noise using the AOGF and BF procedures allow us to assert that the AOGF provides for the Scandinavian small aperture arrays an average SNR gain from 6 to 15 dB in the frequency band 0.2-10 Hz (see table 1).

Fig. 8.2 shows the result of processing by AOGF and BF procedures a weak local event recorded by the A and B rings of the NORESS array (9 sensors). The processing was accomplished in the frequency band 0.1-20 Hz. The AOGF adaptation have been done using noise records at a 100 sec. interval preceding the event *P*-wave arrival (Fig. 8.3). Comparison of the BF (trace 1) and AOGF (trace 2) outputs demonstrates strong the SNR gain due to suppressing coherent NORESS noise by the AOGF. Trace 3 is the seismogram recorded at the central NORESS sensor.

Fig. 8.4 illustrates the AOGF procedure performance for refining surface waves from the Semipalatinsk nuclear explosion (1983-299) obscured by low-frequency noise. The records of 5 long period instruments of the Large Aperture NORSAR array have been processed. Traces 1 and 2 shows the BF and AOGF outputs in the frequency band 0.001-0.1 Hz. The AOGF adaptation has been made using noise records at a 50 min. interval preceding the surface wave arrival (Fig. 8.5). The long period NORSAR sensors, spaced ca 25 km apart, do not exhibit strong noise coherency. Nevertheless, the AOGF output demonstrates that significant improvements in surface wave SNR are obtainable as compared with the BF processing.

Fig 3.6 illustrates the possible sequence of processing procedures seemed to be useful for an explosion detection and its waveform extraction from background of a coda wavws of strong interfering event phase (for a possible scenario for evading a nuclear test ban treaty, see the section 9). We imitated this situation by mixing NORESS records of *P*-wave signal from a Novaya Zemlya (NZ) nuclear explosion (24 Oct. 1990) with records of *P*-wave and its coda from a Hindu Kush (HK) earthquake (25 Oct., 1990: 04.54). The NZ *P*-wave signal was "inserted" in the HK *P*-wave coda starting at 40 sec after the HK *P*-wave arrival. The SNR ratio of amplitude maxima between these event *P*-waves was chosen equal to 0.05. Trace 2 is the output after the NORESS beam steered to Novaya Zemlya in the frequency band 0.1- 5 Hz. The NZ signal is not seen in this frequency band, nevertheless it is reliably detected by the ASOD (trace 1). Note that in this case the detector adaptation was made using the beam trace at the whole time interval being

processed . Trace 3 is the output of AOGF applied to the same data as the BF described above. The AOGF adaptation was made using array records at the time interval 0-40 sec. where only "pure" HK P-wave and its coda are present. Trace 4 is the output of the spatial rejection group filter (SRGF) steered to Novaya Zemlya and adjusted to suppress signals from Hindu Kush. Both traces: 3 and 4 demonstrate the much higher SNR of the NZ *P*-wave in comparison with BF trace 2. Note that the AOGF do not use any information about the arrival direction of the interfering wave but provides better suppression of HK *P*-coda than the SRGF. It can be explained by deviations of the HK *P*-coda wave-front from the plane shape assumed in the SRGF and by the high sensitivity of SRGF performance quality to this assumption. Note also, that in practice the SRGF cannot be easily implemented when an information on interfering event location is absent.

Fig. 8.7 shows the results of detecting and extracting from ARCESS array records the *P*-wave signal from one of the smallest nuclear tests performed at the Semipalatinsk Test Site (28 Dec. 1988). The signal can not be detected in the frequency band 0.2-5 Hz by the conventional BF-STA/LTA method. Neither the adaptive statistically optimal detector applied to the broad-band beamforming output (trace 1) produced detection statistic values reliably exceeding the level of the noise fluctuations (trace 2). In the contrast, this detector applied to the output trace of broad-band AOGF procedure (trace 3) was successful, i.e. produced a strong peak at the time interval of explosion P-wave arrival (trace 4).

The broad-band adaptive statistically optimal detector (ASOD) is a highly sensitive one. It can be used without previous noise suppressing by the BF and AOGF procedures. Fig. 8.8 shows an example of the performance of the ASOD applied to a small local event record made by the central NORESS sensor. Adaptation of the noise whitening filter was made using a 30 sec. interval of preceding noise. Two sharp peaks in the detector statistic trace (trace 2) correspond to *P* and *S*-waves obscured by the noise and hardly seen in the sensor time series (trace 1). Note that the values of these peaks are much greater than the amplitudes of detector statistic fluctuations in the trace 2 during "pure" noise time intervals.

Fig. 8.9 illustrates the performance of maximum likelihood onset time estimator (MLOE) applied to the *S* and *Lg* waveforms of local FINESA event. Trace 4 is the AOGF output containing the *S* and *Lg* waveforms. The onset time likelihood functions were calculated for 3 overlapping time intervals (traces 1, 2 and 3). The first interval contains the *S*-wave signal and its likelihood function (trace 1) has an evident maximum at the "moment" which visually coincides with the S-wave onset time. The second time interval comprises the *S*-wave coda without any pronounced event phase arrival. The likelihood function for this interval (trace 2) is strongly fluctuating without an obvious maximum. The third time interval contains the *Lg*-wave and the likelihood function has a sharp maximum at the “correct” of time.

Fig. 8.10 shows the results of MLOE performance in comparison with the manual onset picking and the conventional SigPro estimation procedure being part of the Intelligent Monitoring System [47,48]. Fig. 8.10a shows the time differences between the automatic MLOE *P*-onsets and the manual picks as a function of SNR for 57 Khibiny Massif Kola events recorded by ARCESS. The two curves at Fig 8.10b. are the cumulative distribution functions of deviations of MLOE and SigPro onsets from manual picks for the 201 first-arriving *P*-phases analysed. For the SigPro (dashed line), 50 % of the onsets are within 0.23 s. of the manual picks, whereas for the MLOE (solid line) the 50% level (median) is as low as 0.005 s.

Fig. 8.11 demonstrates the results of application of the low resolution and the high resolution F-K analysis methods for the estimation of arrival direction of "hidden explosion" wave obscured by interfering event wave (for details see section 9). Fig. 8.11a shows the low resolution F-K map calculated for the mixture of NORESS records of Hindu-Kush (HK) earthquake and Novaya Zemlya (NZ) explosion. The HK and NZ *P*-records were mixed with the relative SNR=1 and the 15 sec. time interval of mixture data was processed by the conventional low resolution F-K analysis procedure. This analysis provides the single maximum at the point close to the midst of two expected slowness vectors. There is no evidence that the data comprise the two different waves. Fig. 8.11b shows the smoothed high resolution F-K map calculated using the same data. The Capon type F-K analysis algorithm incorporating the AR estimate of inverse matrix power spectral density of array data was used. The map has the two explicit peaks that are very close to the correct positions of NZ and HK slowness vectors. The experiment made revealed that the high resolution F-K analysis preserves its resolving power for a SNR as small as 0.3.

Fig. 8.12 shows an application of the conventional F-K analysis and the smoothed high resolution F-K (SHRFK) analysis for the estimation of arrival direction of single seismic wave. The *P*-wave signals from a Hindu Kush earthquake, recorded by the central subarray of large aperture NORSAR array, were processed. In this example, where only 6 sensors placed in the circle with diameter 5 km. were used. The SHRFK analysis with AR-modelling of data exhibits the manifest advantage over the conventional F-K analysis.

Discussion

The results of testing the broad-band adaptive array data processing technique can be regarded as promising. Implementations of this technique for the routine array data analysis could be easily achieved, for example, within the framework of knowledge-based systems such as the Intelligent Monitoring System described in [3,4]. This automated system is mainly based on the conventional multiple narrow-

Table 8.1

Investigation of AOGF SNR gain over beamforming for processing of local and regional event phases

EVENT ORIGIN TIME D:H.M.S	ARRAY, EVENT DISTANCE	PHASE TYPE	AOGF GAIN RELATIVE TO BEAM, Db	AOGF GAIN RELATIVE TO CHANNEL, Db	FREQUENCY BAND, Hz
298:17.51.50	ARCESS 508	P	10.2	12.8	0.2 - 5
282:12.04.13	FINESA 288	P	15.7	17.3	0.2 - 5
		P	13.3	15.0	0.2 - 10
		S	13.6	15.6	0.2 - 5
		S	12.2	13.3	0.2 - 10
		Lg	12.3	13.4	0.2 - 10
294:09.13.00	FINESA 772	P	16.7	18.7	0.2 - 5
		S	15.0	17.7	0.2 - 5
		Lg	14.3	17.1	0.2 - 5
	NORESS 1302	P	12.8	16.9	0.2 - 5
		S	10.6	15.1	0.2 - 5
		Lg	13.5	15.3	0.2 - 5
284:09.38.09	NORESS 1219	P	10.0	13.6	0.2 - 5
		P	6.3	10.1	0.2 - 10
		P	4.3	8.6	0.2 - 20
	FINESA 1771	P	17.1	18.3	0.2 - 5
292:12.31.45	FINESA 164	P	10.3	13.5	0.2 - 5
	ARCESS 390	P	10.4	13.8	0.2 - 10
294:19.32.03	FINESA 259	P	17.5	20.0	0.2 - 10
	NORESS 565	P	11.9	15.8	0.2 - 5
		S	10.0	13.8	0.2 - 5
		Lg	9.7	14.0	0.2 - 5

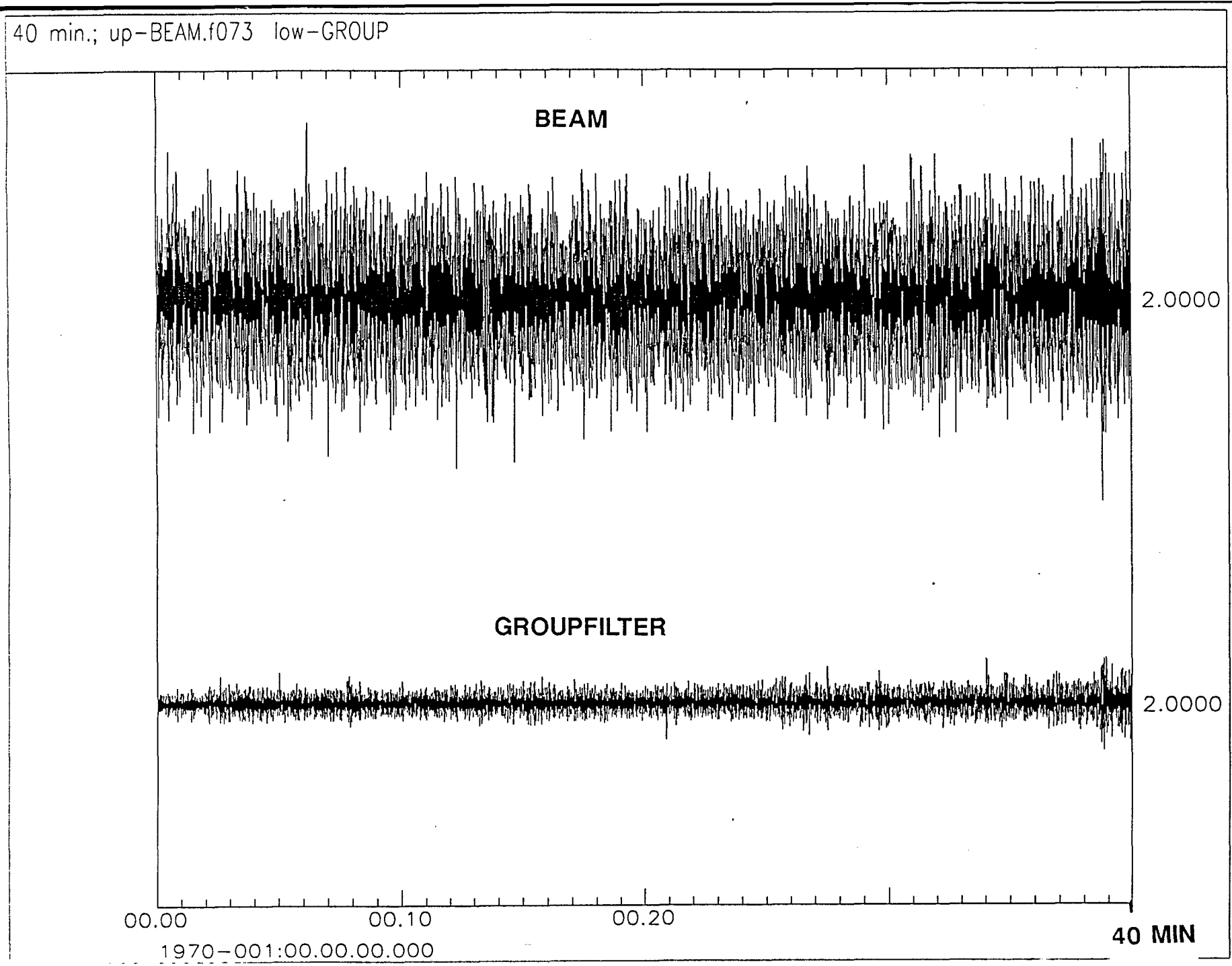

40 min.; up-BEAM.f073 low-GROUP
BEAM
2.0000
GROUPFILTER
2.0000
00.00
00.10
00.20
40 MIN
1970-001:00.00.00.000

Fig. 8.1.

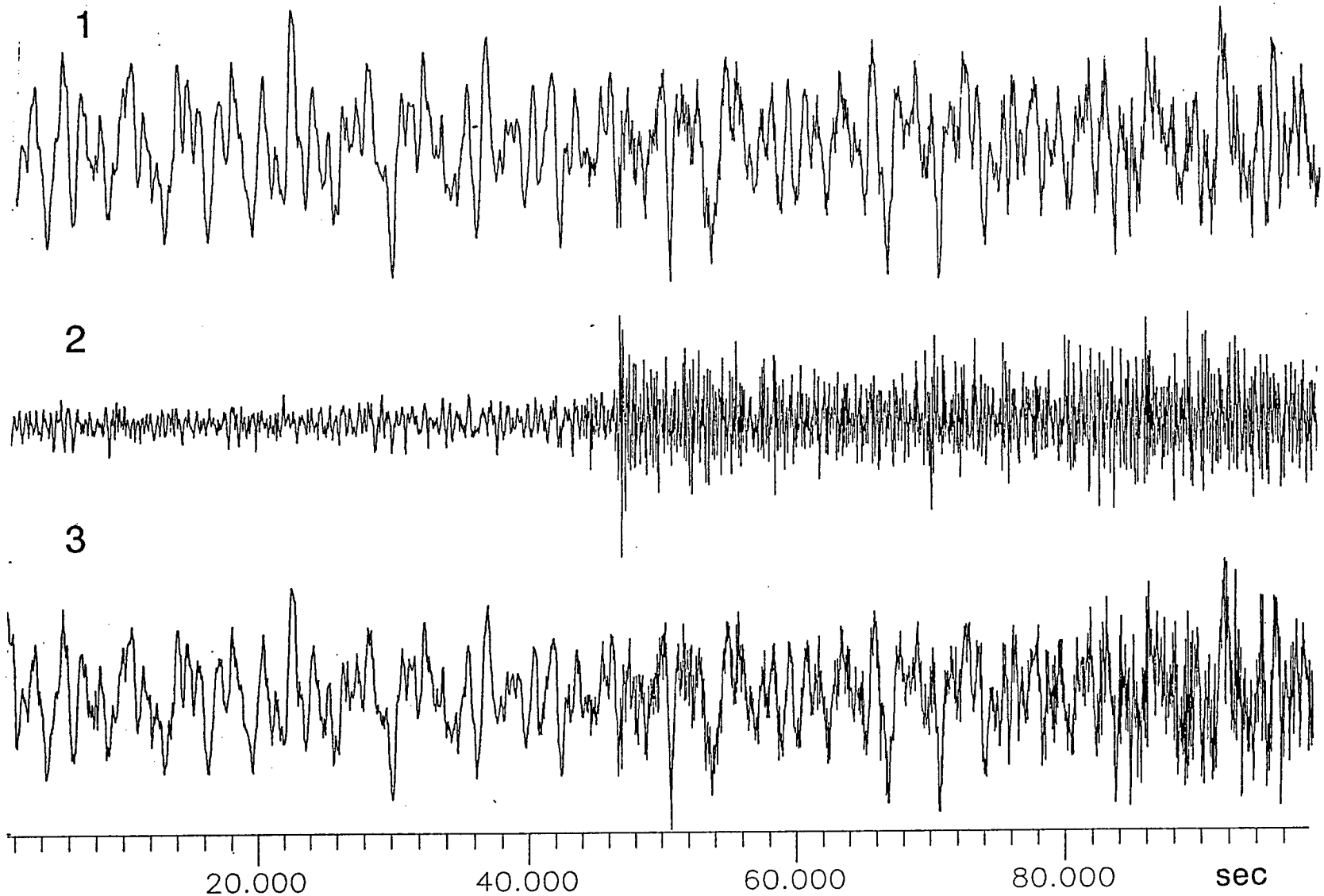


Fig. 8.2.

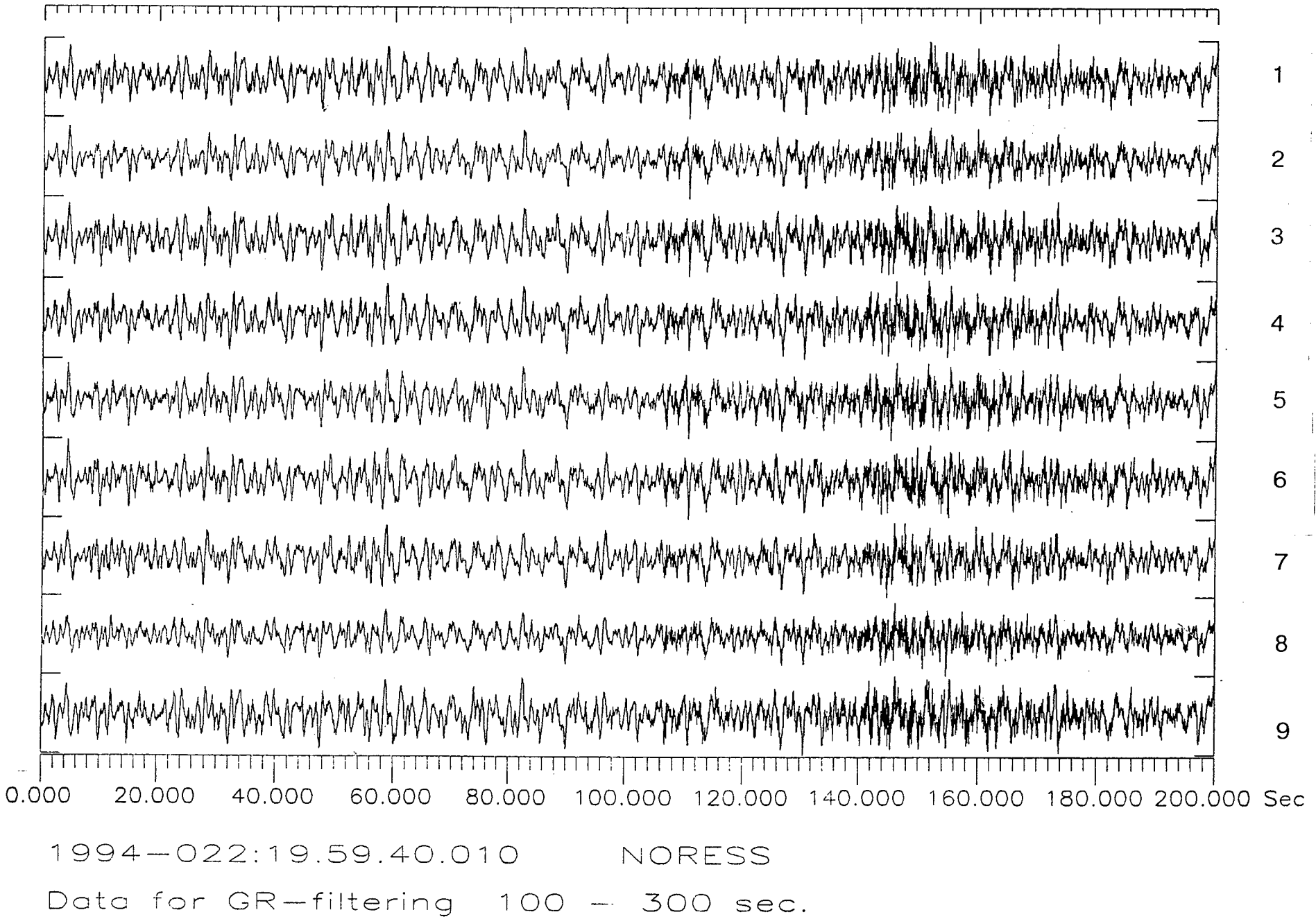

1
2
3
4
5
6
7
8
9
0.000
20.000
40.000
60.000
80.000
100.000
120.000
140.000
160.000
180.000
200.000
Sec
1994—022:19.59.40.010
NORESS
Data for GR—filtering 100 — 300 sec.

Fig. 8.3.

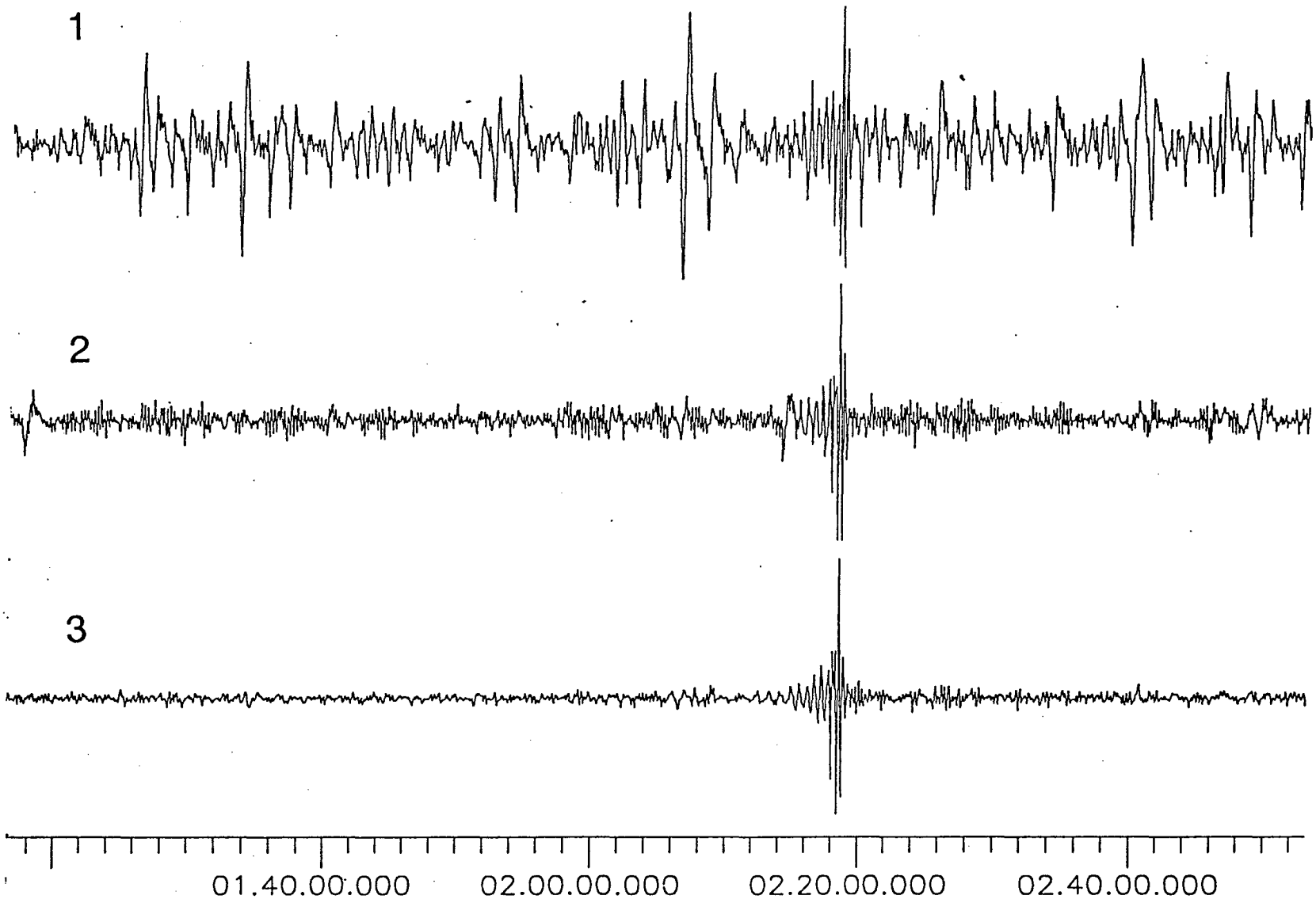


Fig. 8.4.

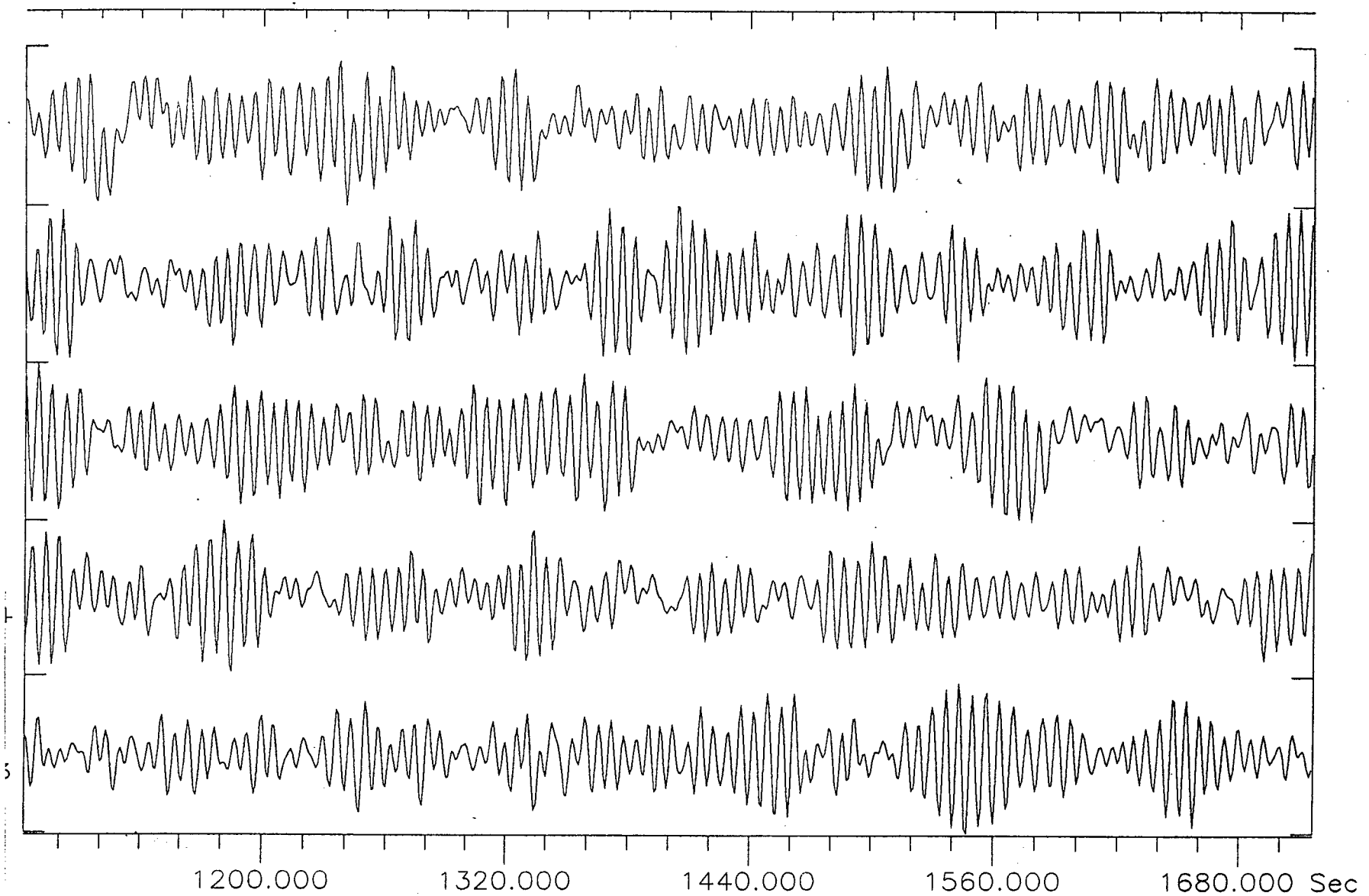

1
2
3
4
5
1200.000
1320.000
1440.000
1560.000
1680.000 Sec
1983—299:01.15.00.000 NORSAR
file 'dsurf.nor3v' Data interval for adptation: 0—3000 sec.

Fig. 8.5.

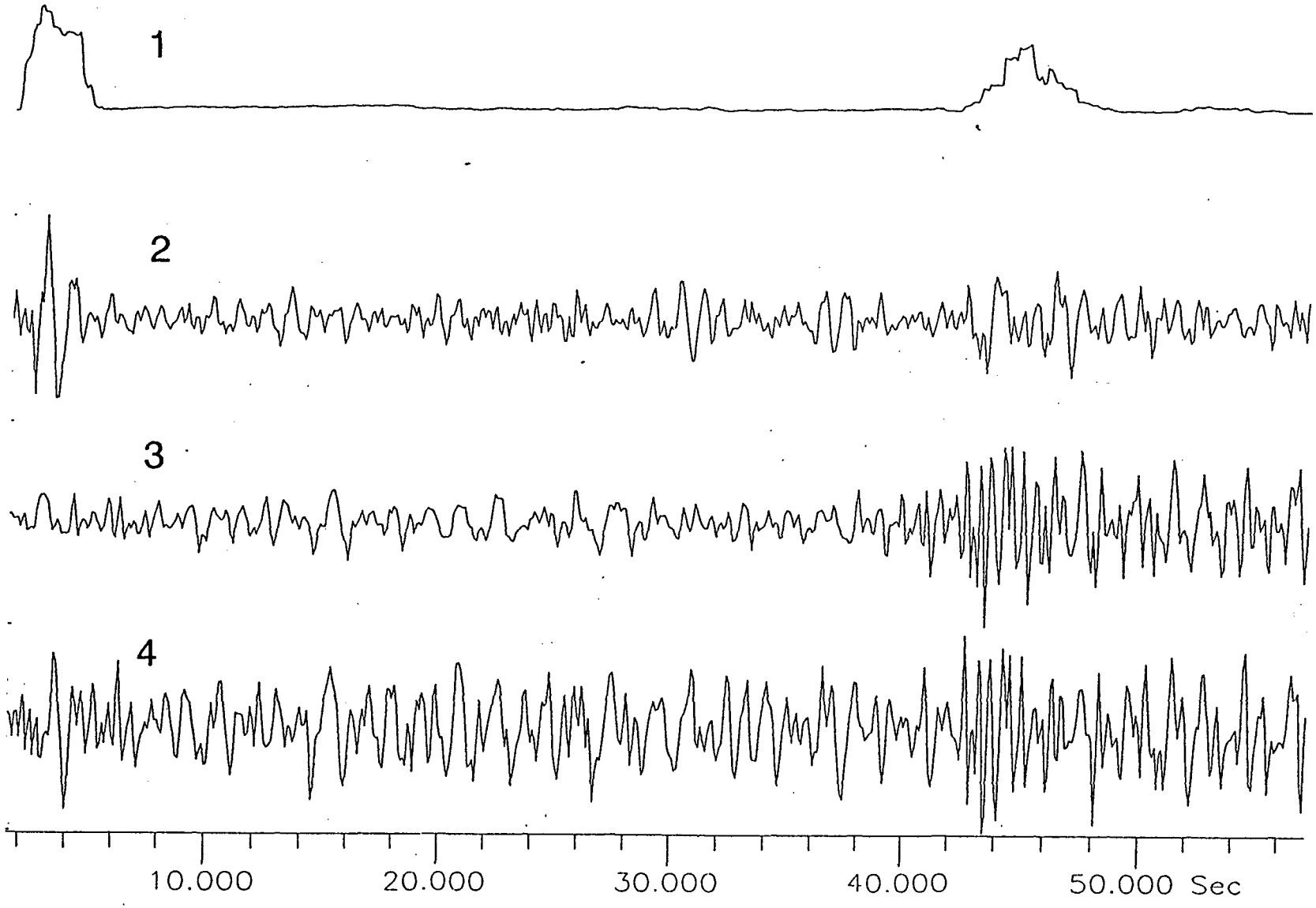


Fig. 8.6.

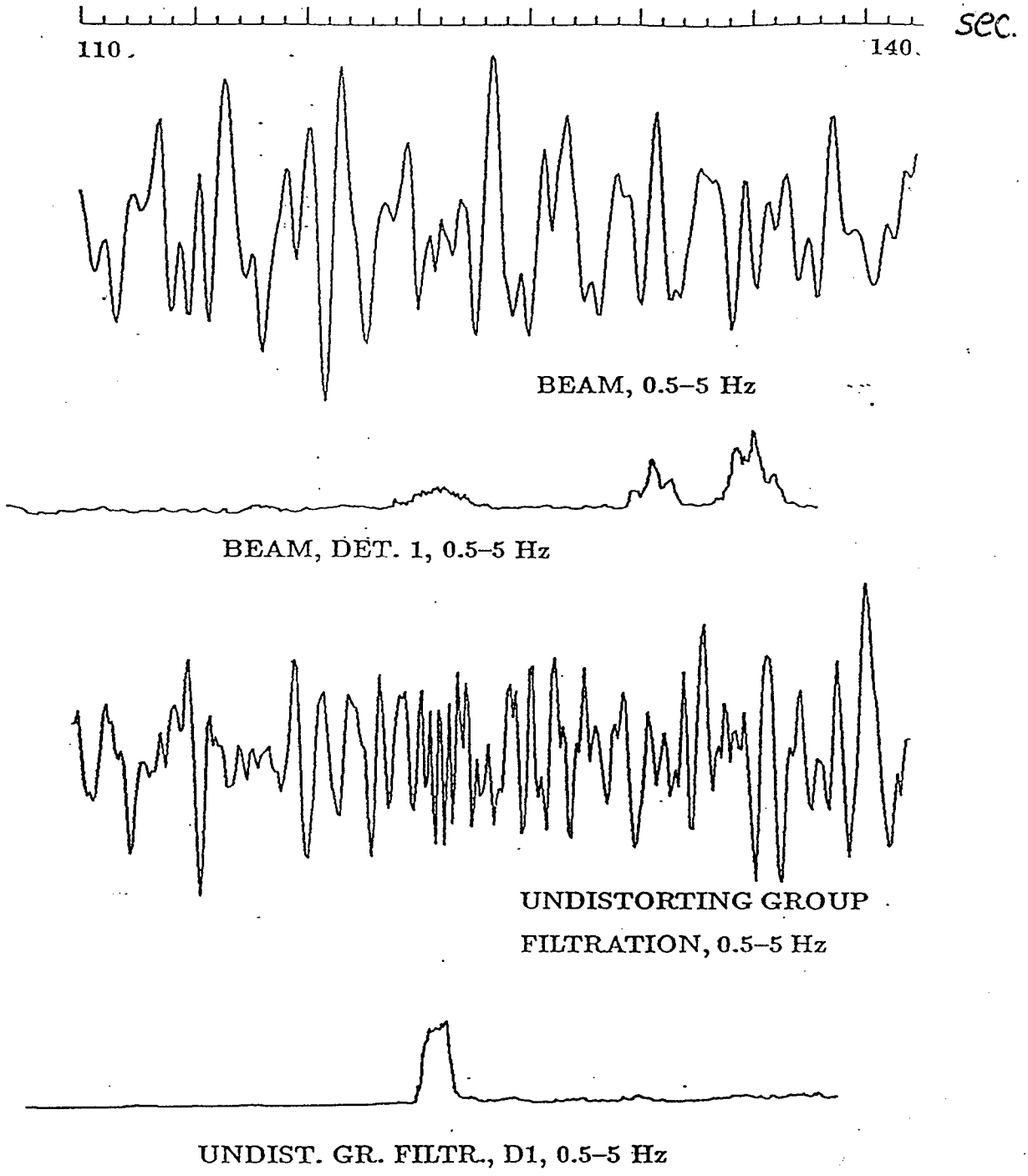

sec.
110.
140.
BEAM, 0.5–5 Hz
BEAM, DET. 1, 0.5–5 Hz
UNDISTORTING GROUP
FILTRATION, 0.5–5 Hz
UNDIST. GR. FILTR., D1, 0.5–5 Hz

Fig. 8.7.

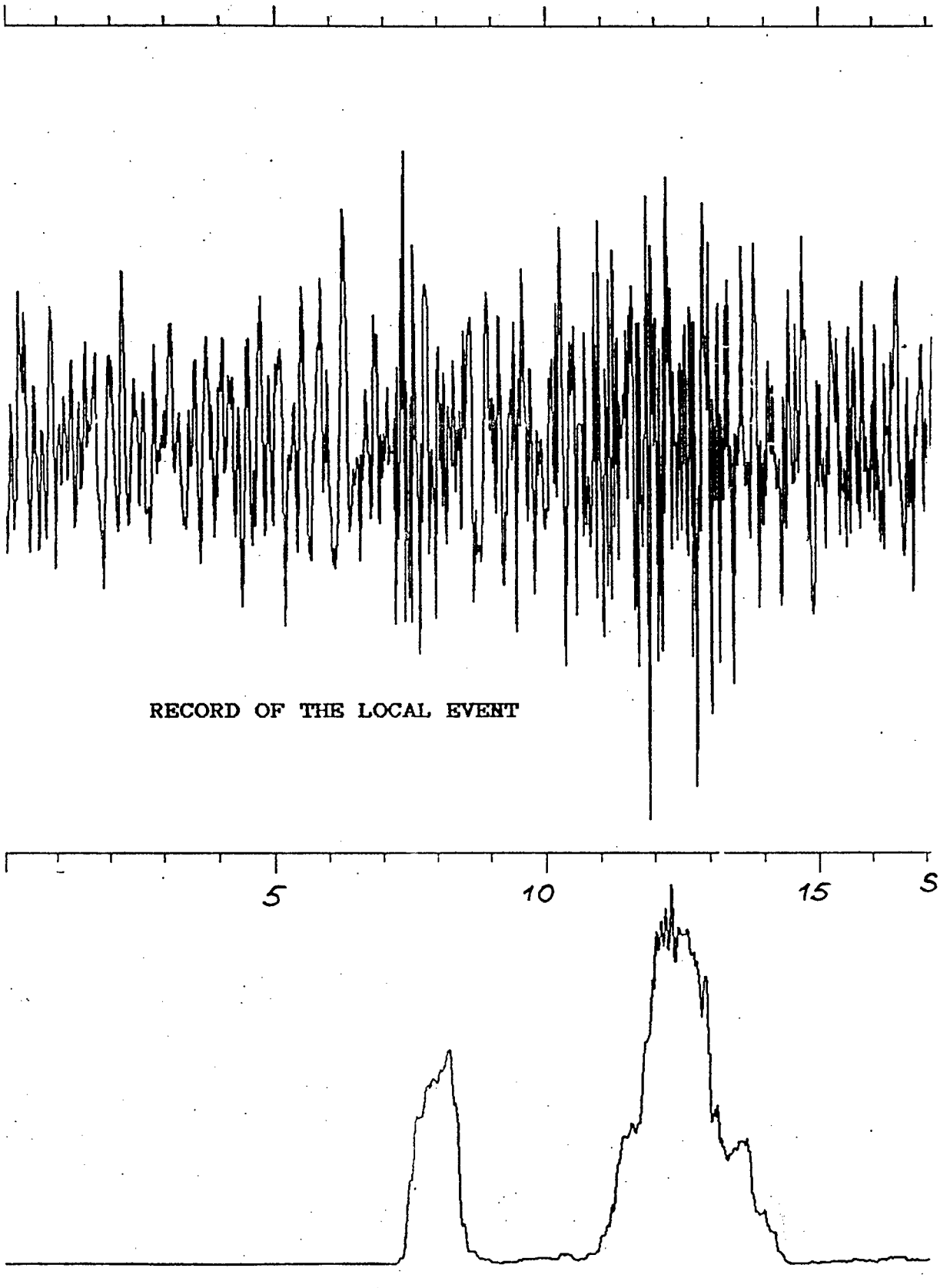

RECORD OF THE LOCAL EVENT
5
10
15
sec
OUTPUT OF DETECTION PROCEDURE APPLIED TO THE RECORD
OF A LOCAL EVENT

Fig. 8.8.

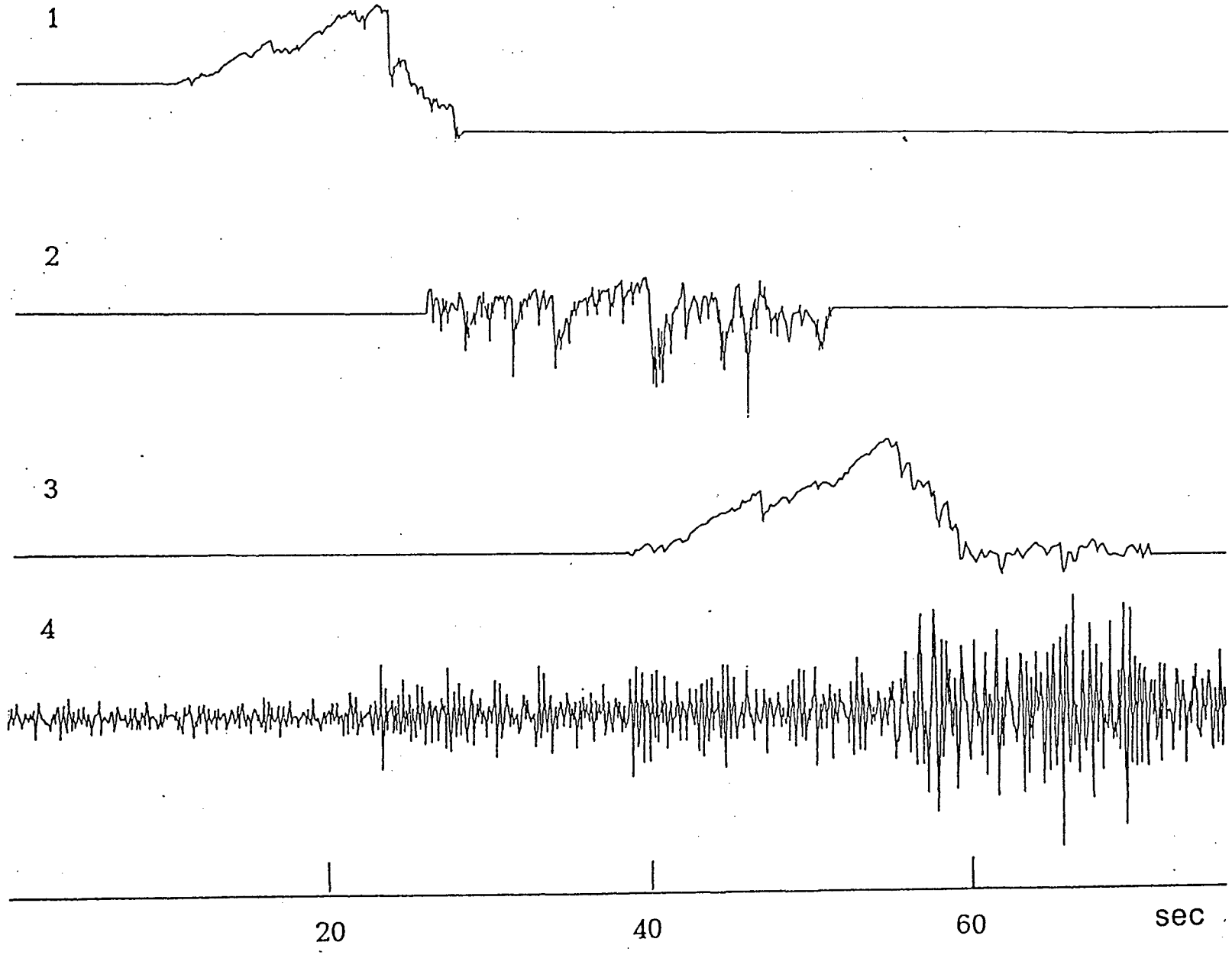


Fig. 8.9.

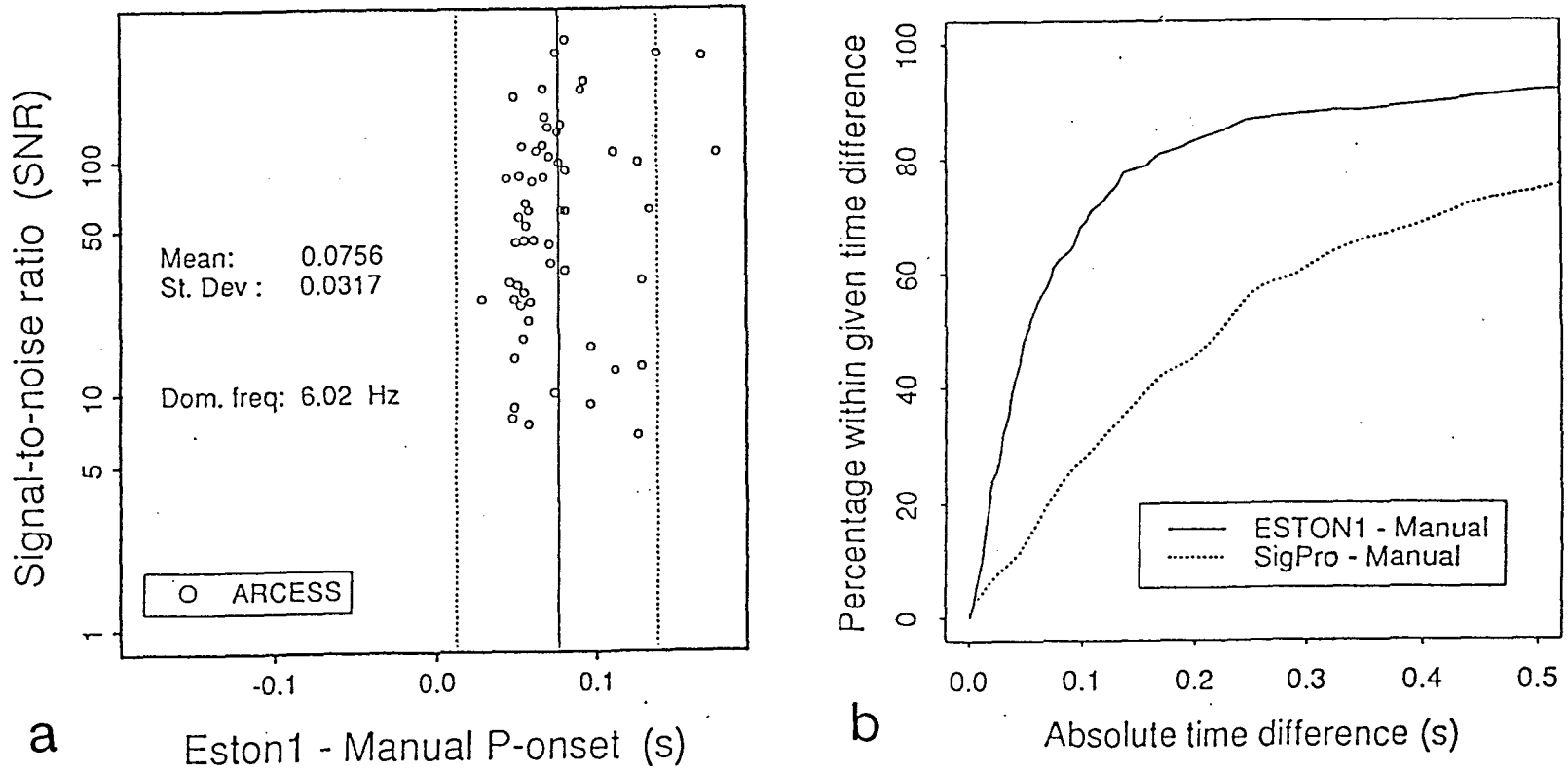

Signal-to-noise ratio (SNR)
1
5
10
50
100
Mean: 0.0756
St. Dev : 0.0317
Dom. freq: 6.02 Hz
O ARCESS
-0.1
0.0
0.1
Eston1 - Manual P-onset (s)
a
Percentage within given time difference
0
20
40
60
80
100
ESTON1 - Manual
SigPro - Manual
0.0
0.1
0.2
0.3
0.4
0.5
Absolute time difference (s)
b

Fig. 8.10.

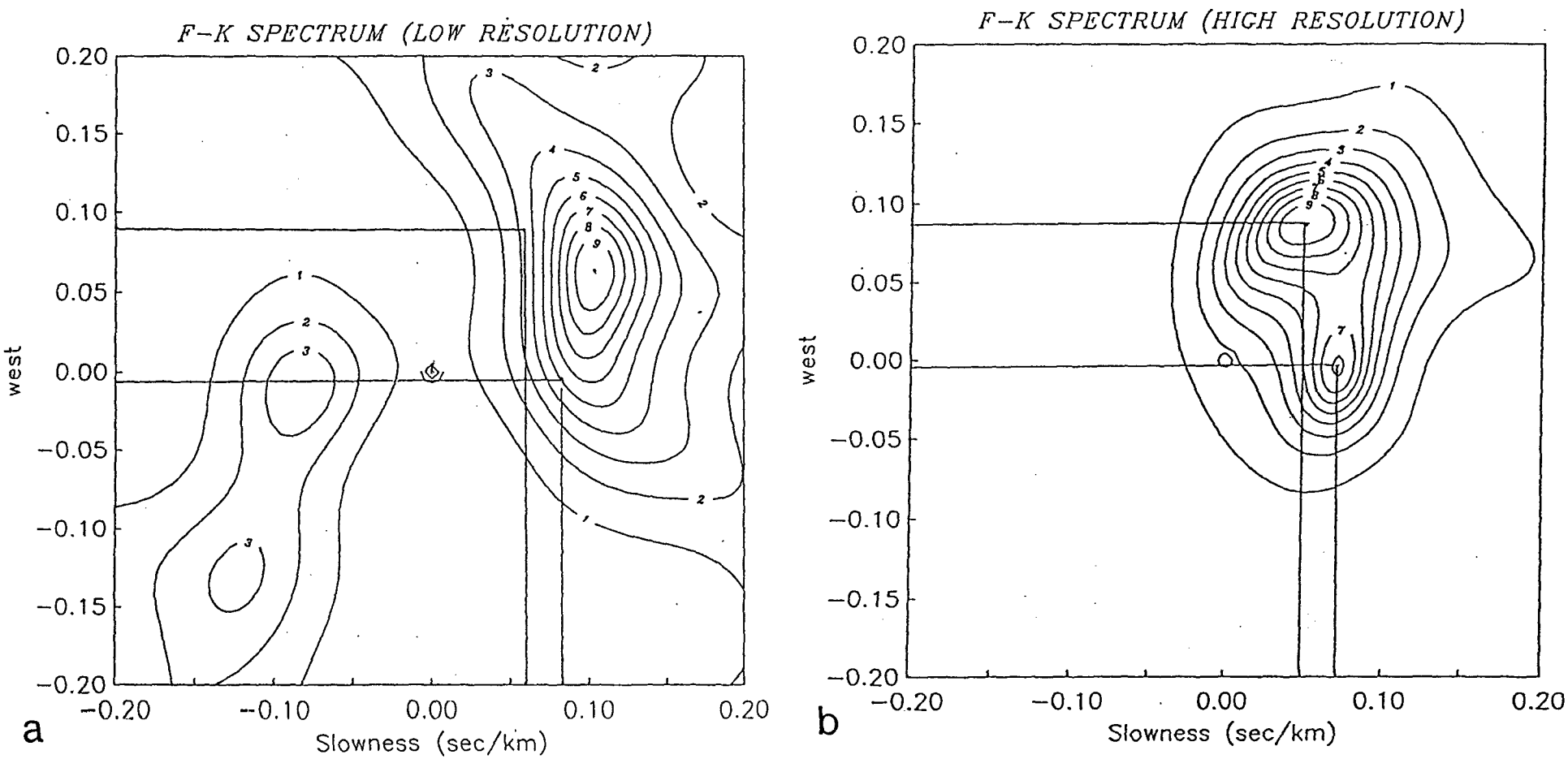

F-K SPECTRUM (LOW RESOLUTION)
F-K SPECTRUM (HIGH RESOLUTION)
west
Slowness (sec/km)
a
b
0.20
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.20

Fig. 8.11.

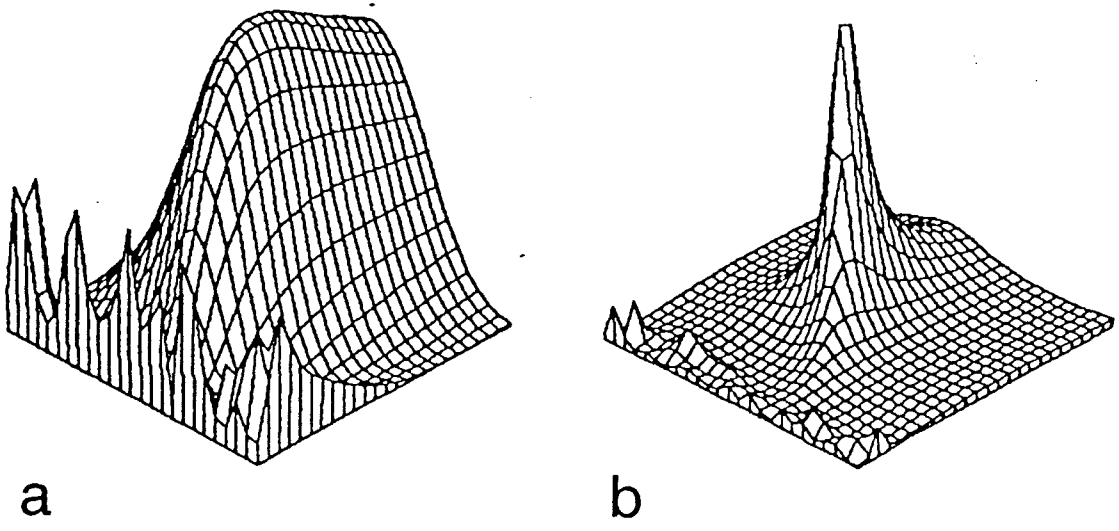


Fig. 8.12.

band beamforming technique. More sophisticated procedures like the adaptive statistically optimal ones could be implemented for recovering missed phases and obtaining more precise signal parameter estimates.

The application of AOGF seems to offer good prospects for recovering low frequency surface waves from regional and teleseismic event. These waves are sometimes obscured by coherent, low frequency noise with the same as for signal frequency band which can be suppressed only by the group filtering. Hence surface phase waveforms can be refined in a frequency band where noise and signal are overlapped. This is also the area for implementation of smoothed high-resolution F-K analysis or adaptive ML-estimation of signal apparent slowness vector. Note that surface waves are very important for estimating yields/magnitudes of weak events [26,59] and for identification of underground nuclear tests.

An another application of AOGF would be exploration of lateral inhomogeneities of the Earth crust beneath seismic arrays. The Detection and location of seismic scatterers can be improved by the implementation of AOGF method modified for this application. The theory and some previous results of testing this approach are given in [44].

The broad-band adaptive optimal detection and optimal on-set time estimation procedures offer an alternative for conventional array data processing techniques and would be valuable as advanced research tool for interactive processing event waveterains with low SNR. Their three-dimensional versions [57,40] could be useful for 3-component seismic data processing on array or stand-alone basis..

9. Detection and parameter estimation of seismic signals obscured by coda of strong interfering events (studies on "hidden" explosion extraction and parameter estimation)

9.1. Introduction

A possible scenario for evading a nuclear test ban treaty is to hide the seismic signals from a clandestine nuclear explosion in the wavetrain of a strong interfering earthquake [45]. The signal detection and parameter estimation of such 'hidden' explosion are the difficult problems if a signal-to-noise ratio is small and frequency contents of explosion and earthquake wavetrains are overlapping. Similar difficulties one encounter in the detection and parameter estimation of weak secondary wave phases obscured by the coda of preceding strong wave phase. These problems manifest the serious challenge especially, if one needs to solve them using data from single observational site. The small aperture seismic array and adaptive statistically optimal data processing provide the effective means to overcome these problems.

The following tasks have to be sequentially solved in the problems mentioned. The first is to detect a signal 'hidden' inside a strong interfering wavetrain. The second is to extract signal waveform from this wavetrain with the purpose to enhance the quality of signal parameter estimation. The third is to estimate the signal parameters such as onset time, spectral content, arrival direction and magnitude.

In general, the above procedures have to be performed under different conditions that arise in different applications:

1) The azimuth and apparent slowness of both the explosion and the earthquake waves are known.

2) The azimuth and apparent slowness of the explosion is known but unknown for the earthquake, e.g., due to uncertainty of the azimuth and slowness characteristics of the earthquake coda or in the case of real time processing.

3) The azimuth and apparent slowness of the earthquake is known, but unknown for the explosion .

4) The azimuth and apparent slowness are unknown both for the explosion and the earthquake.

In practical applications, the scenarios 1 and 3 above would probably be the most useful. In these two cases, it is assumed that the earthquake has been well located, so that azimuth and apparent slowness can be assumed known. The search for a potential hidden signals (the explosion) might be done without any assumptions on its location (case 3), or one might want to focus upon a site of special interest (case 1).

The scenarios 2 and 4 are most interesting for detecting potential hidden events in the real time mode and if their signals are received relatively far in the coda of the earthquake. Even if the earthquake location is known, the scatter in azimuth and slowness of coda waves might make it appropriate to apply the model of an unknown earthquake source, as done in these cases.

In Section 9.2 we discuss array data processing algorithms that seem to be helpful to solve the problem under condition 1) and 2). At Section 9.3 we tackle the problem under 3) and 4) conditions. In cases 1) and 2) the apparently best way to detect and estimate an explosion signal waveform is to apply some group filtering procedures by which the array spatial receiver function attains its maximal gain in the azimuth and slowness of the explosion while minimising the gain in the azimuth and slowness of the interfering earthquake.

9.2. Signal detection and waveform extraction in the coda of a strong interfering event

9.2.1 Spatial Rejection Filter performance.

When the azimuth and slowness of both the explosion and earthquake are known we propose to use the spatial rejection group filter (SRGF) to improve the detection and parameter estimation capability of the 'hidden' explosion. Let us remember that a group filter (GF) $\phi(f) = (\phi_1(f),...,\phi_m(f))^T$ is a multichannel filter which transforms the m input array data traces $x(f) = (x_1(f),...x_m(f))^T$ (T is the sign of transposition), into scalar trace $y(f)$ via equation:

$$y(f) = \phi^*(f)\, x(f), \qquad (1)$$

where f is a frequency, $0 < f < f_{smp}/2$, f_{samp} is the sampling frequency and * denotes the complex conjugation. The vector frequency response $\phi_r(f)$ of the spatial rejection group filter was derived in Section 3 (eq.(3.13)):

$$\phi^*_r(f) = [h^*(f, p)\, \boldsymbol{B}(f)] / [(h^*(f, p)\, \boldsymbol{B}(f)\, h(f, p)] \qquad (2)$$

where the rejection matrix $\boldsymbol{B}(f)$ is calculated in accordance with eq.(3.14) in which $q(f) = (q(f)_1,..., q(f)_m)^T$ is a vector frequency response of the media beneath the array along propagation paths for the interfering ('noise') wave between k-th, $k \in 1,m$ and 1-st sensors; $h(f,p)$ is the same for the signal wave with a vector apparent slowness p.

For the problem under discussion the 'residual beamforming' (RB) method is often used [8,23]. By this method the beam composed from initial data and steered to the 'noise' direction is subtracted (after proper delays) from every array trace to create residual traces. Then a new beam steered to the signal direction is composed from the residuals. This method can be formulated as the implementation of group filtering procedure by eq.(1) with vector frequency response

$$\phi_{br}^*(f) = h^*(f, p)\, \boldsymbol{B}(f)\,. \qquad (3)$$

As it is shown in Section 3.3 both the group filters by eq.(2) and eq.(4) suppress (theoretically - completely eliminate) the interfering purely coherent wave arriving from the assigned direction. This capability is due to the special structure of matrix $B(f)$ explained in Section 3.3. It is easy to show that the vector time series $u(f) = (u_1(f),...,u_m(f))^T$ produced by the matrix spatial residual filter:

$$u(f) = \boldsymbol{B}(f)\, x(f). \qquad (4)$$

does not contain the interfering plane wave arriving from proper (correct) direction.

Nevertheless the RB method has a serious disadvantage because it distorts the frequency content of the explosion signal and therefore can not be regarded as a 'pure spatial' filtering. In contrast the SRGF is a signal undistorting procedure, i.e. if a signal purely coherent wave arrives from the direction assigned by steering vector p the output trace of the filter eq.(2) coincides with a signal waveform.

To illustrate the difference in performance between the spatial rejection group filtering and the residual beamforming methods we have simulated a mixture of two fully coherent plane waves with different azimuths, apparent slowness and waveforms using the NORESS array sensor coordinates. The signal plane wave was assumed arriving from the Novaya Zemlya Test Site (azimuth = 32.9 degrees, apparent velocity = 10.4 km/sec) and the signal waveform was generated as a linear frequency modulated cosine signal with a constant amplitude ('sweep signal'). The interfering (noise) plane wave was simulated arrive from the Hindu-Kush area (azimuth =101.4 degrees, apparent velocity = 14.8 km/sec) and its waveform was set equal to the beam waveform of the real NORESS records of Hindu-Kush earthquake P-phase. The ratio between the maximum amplitudes of signal and interfering waves was 0.13.

Fig.9.1 shows the residuals after processing simulated data with the matrix spatial rejection filter eq.(4). We see that the interfering wave is completely suppressed but the shape of 'hidden' signals is distorted at low frequencies. We also note, that distortions are different for different channels.

The upper trace in Fig.9.2 is the result of conventional beamforming (BF) procedure steered to the Novaya Zemlya Site. In this trace the low SNR hidden signal can not be distinguished. The second trace is the result of RB method realised as the group filtering in the frequency domain in accordance with equations (1) and (3). The interfering earthquake waveform is completely suppressed, but the 'hidden' signal amplitudes are strongly reduced at low frequencies. This example shows that RB procedure suppresses the signal low frequency components. It follows from eq.(3) and (3.14), (3.18) that the signal frequency distortions are dependent on both the array geometry, and the signal and noise arrival directions and hence are difficulty predictable. The third trace in Fig.9.2 is the output of spatial rejection group filter by eq. (1), (2). This group filter extracts the 'hidden' signal from the interfering wave and retains its sweep waveform undistorted. The fluctuations of signal amplitude at the end port of the trace are explained by signal frequencies approaching to the Nyquist frequency of data.

In practice, the explosion and signal waves do not consist of a single plane waves component i.e. they are not fully coherent. To illustrate the performance of the SRGF and the RB under such real conditions we have created a mixture data by imposing down-scaled real NORESS records of the Novaya Zemlya (NZ) nuclear explosion of 24 October 1990 into the *P*-wave coda of NORESS records of the Hindu-Kush earthquake (HK)(origin time: Oct. 25, 1990, 04.53.59.9). The ratio between the maximum amplitudes of the explosion and the earthquake was 0.2 and the P-phase from the explosion was set to at 12 s after the HK P-arrival. The lower trace (4) in Fig.9.3 shows the event mixture at the central element of NORESS array. The top trace (1) is the conventional beam steered to the NZ site. Also in this case we can see that the output from RB procedure (trace 3) contain less low

frequency components for the hidden signal as compare to the output from the SRGF (trace 2). This is not so important in the example examined where the high-frequency P-wave from nuclear explosion being extracted. But for extracting low-frequency signals from surface waves to the implementation of SRGF is much preferable in comparison with the BR method. Note also that the high pass filtering of the SRGF output (with the controllable rejection of low frequency components) provides the resulting SNR higher then the SNR of the RB output.

9.2.2. Adaptive Optimal Group Filtering.

The late part of an earthquake coda consists of wave components arriving from very different azimuths and sownesses for rejecting this type of interfering energy. It is not very meaningful to use the spatial rejection filter. The same is applied to real time processing, when the time delays (and phase shifts) of array signals of interfering event are unknown. In this situation (case 2 of Introduction) the adaptive optimal group filter (AOGF) can be helpful [45] . The vector frequency response of AOGF procedure is

$$\phi^*_o(f) = [h^*(f,p)\ \hat{F}^{-1}(f)] / [h^*(f,p)\ \hat{F}^{-1}(f)\, h(f,p)] \qquad (5)$$

where $\hat{F}^{-1}(f)$ is an estimate of inverse matrix power spectral density of array noise (in our case 'noise' is an interfering wave). It was noted in Section 3.2 (see also [ ]) that theoretically the time interval of data used for optimal group filter adaptation can include the *plane wave* signal we intend to retrieve and this not leads to degrading the performance of AOGF filtering. Additional requirement is that the group filter must be steered accurately to the signal wave arrival direction. We will refer this type of adaptation as self-adaptation.

Fig.9.4 presents the results from processing the mixture of simulated plane waves (described in connection with Fig.9.1 and Fig.9.2) using the self-adaptive AOGF. The upper trace (1) is the result of conventional beamforming steered to the Novaya Zemlya Test Site. The second trace is the output from processing with the self-adapted AOGF and the third trace is the output from the SRGF (assuming signal from Novaya Zemlya and interfering energy from Hindu Kush) The bottom trace (4) shows the simulated mixture as observed on the central NORESS sensor. When comparing the second and third traces we find that the self-adaptive AOGF in the case of two interfering *plane* waves performs almost as well as the spatial rejection group filter, and we emphasise that implementation of self-adaptive group filter does not require any information on the arrival direction of the interfering event and onset time of the 'hidden' signal.

However, our experiments with mixtures of real events revealed the significant reduction in the performance of the AOGF filtering when the self-adaptation was performed as compared to adaptation to the interfering event made using 'pure

noise' records. We believe that it is due to the fact that both the signal and the interfering waves deviates strongly from single plane wave components. The following figures illustrates this problem.

In Fig.9.5 the adaptation of the AOGF was made using 'pure' Hindu Kush earthquake recording at the whole time interval. The NZ signal was then mixed with this recording (SNR 0.1, signal onset time 12 sec after HK P-wave arrival). The results of AOGF filtering are shown in traces 2 and 3 for two slightly different versions of the algorithm. For comparison, the outputs from the conventional beamforming and SRGF procedures are shown in traces 1 and 4. For this artificial situation, the performance of AOGF is excellent. At the same time, the BF and SRGF procedures failed to extract the 'hidden' NZ explosion signal with SNR 0.1.

However, results from AOGF after self-adaptation do not give nearly as good results, see this in traces 2 and 3 of Fig.9.6. The self-adapting AOGF procedure (traces 2 and 3) provides the same bad extraction of the 'hidden' NZ signal as the SRGF (trace 4), both the procedures practically failed to retrieve the signal from the mixture with small SNR 0.1.

In Fig.9.7 the SNR of the NZ signal was rised to 0.3, and we see that both the outputs from AOGF after self adaptation (traces 2 and 3) and from the SRGF (trace 4) clearly extract the 'hidden' NZ signal from the earthquake coda.

9.2.3 Adaptive Statistical Phase Detection.

The difference in frequency content between the Novaya Zemlya explosion and the Hindu Kush earthquake may enable us to detect the explosion on the self-adapted AOGF trace even if the SNR is very small. Fig.9.8 shows the result after processing the self-adapted AOGF output with the adaptive optimal phase detector (AOPD) described in Section 4. The SNR of the NZ signal was set 0.05, and the NZ P-phase arrived - 21 sec after the Hindu Kush P-wave arrival. The AOPD algorithm includes the adaptation procedure and the procedure for phase detection in a moving window. The entire self-adapted AOGF output (trace 3) was used for detector adaptation, resulting in an averaged AR-model of the trace. Then the AR-coefficients are used for calculation of detector statistic [36,41] using data within 4-sec window moving along the self-adapted AOGF trace. The detector statistic values are shown in the upper trace. We see that althow the explosion signal is not clearly identified on the output from the self-adapted AOGF (trace 3) or SRGF (trace 4), the output from AOPD (trace 1) gives us convincing evidence of the presence of the signal.

The high sensitivity of the AOPD can therefore be helpful as a tool to be used for detection and waveform extraction of signals with low SNR. A practical procedure would be first to run the AOPD on a conventional beam output with the purpose to detect the signal and to determine the approximate onset time of the

signal. The second step would comprise the AOGF adaptation using the interval before (and may be after) the signal arrival. The final step would be to process the entire data segment with the AOGF. The testing of this sequence of processing procedures using a mixture of the Novaya Zemlya explosion and Hindu Kush earthquake NORESS records was described afore section 8 (see Fig.8.6).

9.3. Estimation of azimuths and apparent slownesses of interfering seismic waves

9.3.1. Estimation of spatial spectrum of interfering waves by high-resolution methods.

Under the conditions 3) and 4) of Introduction the detection and parameter estimation of hidden explosion signal have to be performed without any information on azimuth and apparent slowness of a suspected event. This makes almost impossible the implementation of AOGF and SRGF because these methods imply the group filters to be steered in the direction of hidden event. One can try to use in this case a 'fan' of such group filters steered in some set of directions, but this is extremely time consuming and do not provide the needed accuracy. More promising approach is to reveal the fact that array observations are composed by waves from two different events and then to estimate the azimuth and apparent slowness of hidden event. To perform this one can implement different methods of high resolution F-K analysis and statistically optimal estimation of apparent slowness vector described in Sections 6. At the next processing step the group filtering and adaptive phase detection procedures can be used for signal waveform extraction and estimation of the hidden event onset time, frequency content and magnitude.

The most difficult case is the case 4), where azimuths and sownesses are unknown neither for the interfering nor for the hidden events. In this case, apparently, the only one way exists: to estimate the spatial spectrum of seismic field based on the array observations i.e. to evaluate a seismic field F-K map. The presence of two explicit peaks at the map testifies that the array data are composed by seismic waves from two events. The apparent slowness vectors for which maximal values of these peaks are attained, give the estimates of azimuth and slowness of the events.

To compare capabilities of different F-K analysis methods for this problem we made experiments with mixtures of real NORSAR records from the Novaya Zemlya (NZ) explosion and the Hindu-Kush (HK) earthquake. We used also simulated NORESS records to form mixtures of several plane waves arriving to NORESS from different directions. We tested in these experiments several methods of high resolution (HR) F-K analysis along with the conventional broad-band F-K analysis [55]. The object of our primary interests was the known Capon-type HR algorithm which uses some estimate of inverse matrix power spectrum density (MPSD) of

array records. The Capon-type F-K spectrum is calculated in accordance with eq.(6.1). Different versions of the Capon algorithm are discerned by methods used for inverse MPSD evaluation. We implemented for this purpose the multichannel ARMA-modelling of array records, and also tested the conventional Barttlet MPSD estimate [13] with subsequent its inversion at every frequency. Both the AR-modelling and the Bartlett method provide an inverse MPSD estimate $\hat{F}^{-1}(f)$ smoothed over frequencies. So, being substituted in eq.(6.1) they provide different variants of the smoothed HR F-K analysis. We will refer them below as the AR-HR and B-HR methods of F-K analysis.

As a version of the conventional broad band F-K analysis we tested the F-K spectrum estimate as following

$$\hat{P}_{lr}(f, p) = h^*(f, p)\ \hat{F}(f)\ h(f, p), \qquad (6)$$

where $\hat{F}(f)$ is the Bartlett MPSD estimate. Due to smoothness of the Bartlett MPSD estimate over frequencies the F-K spectrum estimate eq.(6) can be regarded as a broad-band one. We refer it below as the Batrlett low-resolution (B-LR) F-K analysis.

A computer simulation was performed to assess a capability of the AR-HR and B-LR algorithms to resolve plane waves arriving to the NORESS array with close azimuths. Mixtures of plane wave records with random-shape waveforms have been simulated. The power spectra of simulated waveforms was the same as for the NZ explosion power spectrum. The F-K spectrum maps resulting from these experiments are shown in Fig.9.9 - Fig.9.11. Fig.9.9,9.10a,b show the 3-dimensional plots of F-K spectra by the B-LR and AR-HR F-K analysis for the simulated mixture of 6 plane waves. These waves have identical powers and the some apparent slownesses 0.34 sec/km but the different azimuths which equidistantly cover the entire circle. We see that for the NORESS geometry the B-LR method (Fig.9.9) gives a false lobe at the centre of the F-K spectrum map. We see also that the peak from the wave with the slownesses (0.0, -0.240) is heavily suppressed. The AR-HR method (Fig.9.10) provides the narrow signal peaks without a false lobe at the central part of the map.

Fig.9.11,9.12 show the F-K spectra estimated by the B-LR AR-HR methods for the simulated mixture consisting 12 plane waves with identical powers and the apparent slownesses 0.34 sec/km but the different equidistantly distributed azimuths. We see that the B-LR estimate (Fig.9.11) creates a false central lobe and misses the three waves (with numbers 2, 3 and 11 clockwise from the peak with (p_x, p_y) coordinates *(0.0, -034)*). The AR-HR estimate (Fig.9.12) allows to detect all the 12 waves but assigns them the different intensities. The latter disadvantage is the well-known specific feature of all the high-resolution F-K analysis methods. In Fig.9.13 we display (mostly, as curios example) the AR-HR estimate of F-K

spectrum for the 18 simulated plane waves with characteristics in similar ones as in the previous case. We see that the estimate enable us to detect all the waves.

The next experiment we made with the real NORESS records from the Novaya Zemlya (NZ) explosion and the Hindu Kush (HK) earthquake described in Section 9.1. We composed the artificial mixtures of this records with SNR 1.0 and 0.3 and with the onset time of NZ explosion P-wave 10 sec. after HK P-wave. The estimates of F-K spectra of these mixtures were calculated using the five F-K analysis methods. Three of them: the B-LR, AR-HR and B-HR algorithms are described above. Besides we tested on this data the two modern methods currently used in radar and sonar applications for detection and parameter estimation of several interfering plane waves. These are so called MUSIC method and Eigen-Vector method [38]. We will use below the abbreviations for them as the M-HR and E-HR estimates

The estimates of F-K spectrum for the mixture with SNR=1.0 are shown in Fig.9.14-9.19 and for the mixture with SNR=0.3 - in Fig.9.17,9.18. The B-LR estimate do not enable one to reveal the presence of two waves with different azimuths and slownesses already in the case SNR=1 (Fig.9.14). All the high resolution F-K analysis algorithms implemented in the experiment produced the F-K spectrum maps (Fig.9.15-9.17 for SNR 0.1 and Fig.9.18, 9.19 for SNR 0.3) that allows to detect the two waves arriving in the same time interval from different directions. One see at these maps the distinct peaks corresponding to the azimuths and slownesses of NZ and HK events. The estimates of these events slowness vectors (p_x, p_y) that were derived from the maps as coordinates of the peaks maxima, are rather consistent (Table 9.1). For the AR-HR, B-HR and M-HR algorithms the deviations of these estimates from the real (p_x, p_y) values are less than 15%. These three algorithms provides at the F-K spectrum maps the peaks with approximately the same levels for both the waves. This is so even in the case SNR 0.3 where the NZ explosion wave has at 3 time less power than the HK earthquake wave. The E-HR algorithm manifested in our experiments slightly worse performance than the other high-resolution algorithms. It gave the largest deviations of slowness vector estimates from the real values and provided very different maximal values of the wave peaks even in the case SNR=1. Note, that though the AR-HR, B-HR and M-HR methods seems to have similar resolving power we regard the AR-HR as preferable method because it is the least time consuming.

9.3.2. Estimation of slowness vector of signal wave obscured by interfering wave with known direction.

If the azimuth and a slowness for interfering wave are known a'priory or have been estimated via processing of array data containing "pure" noise records, some more precise algorithms then F-K analysis can be proposed for the signal wave

azimuth and slowness estimation. The statistical approach that can be helpful for design of such algorithms is described in details in Sections 6 and 7. The maximum likelihood estimate of slowness vector p for this case is given by eq.(6.2). It have been noted in Section 6 that for the case being discussed the F-K analysis of "beam residuals" is often used. Equation (6.3) gives the mathematical expression for this method. The procedure for high resolution F-K analysis of "beam residuals" is explained too. We tested the performance of "beam residuals" method for the estimation of slowness vector of the Novaya Zemlya explosion 'hidden' in the coda of Hindu Kush earthquake. The same mixture $x(f)$ of this event NORESS records was used as while testing the F-K analysis methods (SNR = 1). At the first step the residuals $y(f)$ from the beam steered to the azimuth and slowness of HK earthquake were calculated with the formula: $y(f) = \boldsymbol{B}(f)\, x(f)$, where matrix $\boldsymbol{B}(f)$ depends on the time delays of HK *P*-wave signals in NORESS sensors. The residuals $y(f)$ then were processed by the low-resolution and high resolution F-K analysis procedures. It is easy to shown that the low resolution F-K spectrum estimate by eq.(6) is very close to the conventional "beam residuals" F-K spectrum estimate given by eq.(6.3) and even is preferable from the statistical point of view. Fig.9.20, 9.21 show the low resolution and high resolution F-K estimates for the residuals. The both procedures produced the maps having the single peak. The peak maximum locations at both the maps are rather far from the real value of NZ event slowness vector (which is also shown at the figures). Both the beam residuals F-K analysis algorithms provide the worse results then the high resolution F-K analysis algorithms applied to the initial mixture of NZ explosion and HK earthquake data.

Table 9.1

Slowness estimates for different high resolution methods

	Real AR	B-HRFKA	A-HRFKA	M-HRFKA	E-HRFKA
SNR=1					
NZEX Px	0.052	0.051	0.048	-	0.038
NZEX Py	0.081	0.086	0.077	-	0.065
HKE Px	0.066	0.072	0.058	-	0.054
HKE Py	-0.013	-0.007	-0.008	-	-0.005
SNR=0.3					
NZEX Px	0.052	0.049	-	0.056	-
NZEX Py	0.081	0.091	-	0.089	-
HKE Px	0.066	0.070	-	0.062	-
HKE Py	-0.013	-0.006	-	-0.017	-

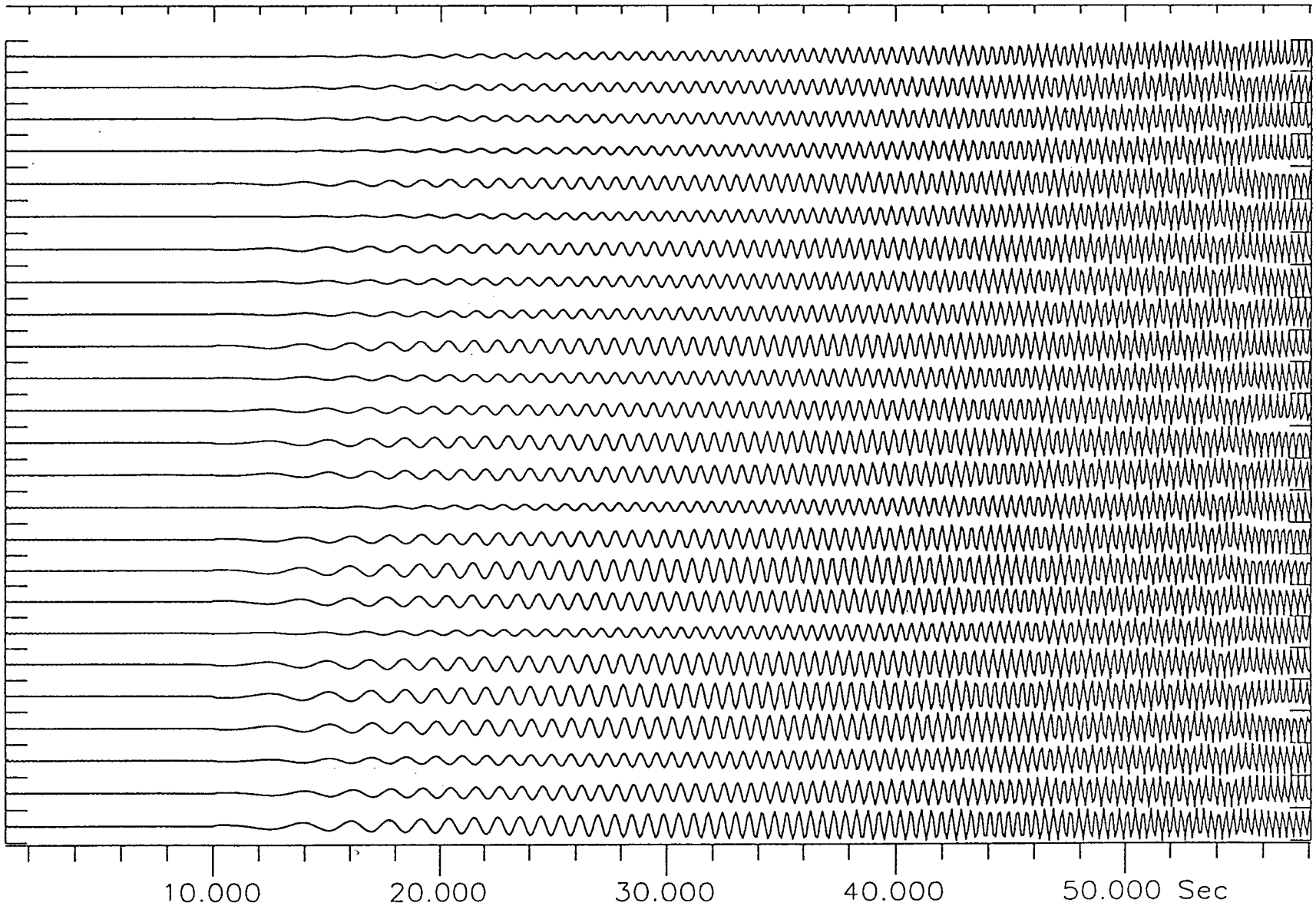

10.000
20.000
30.000
40.000
50.000 Sec
1990−297:15.02.25.711
NRS_sz
Lf= 0 Hf= 5 .
NORESS MODELS: Mixture of Hindu +NZ signals, SNR= 0.130 .

Fig. 9.1.

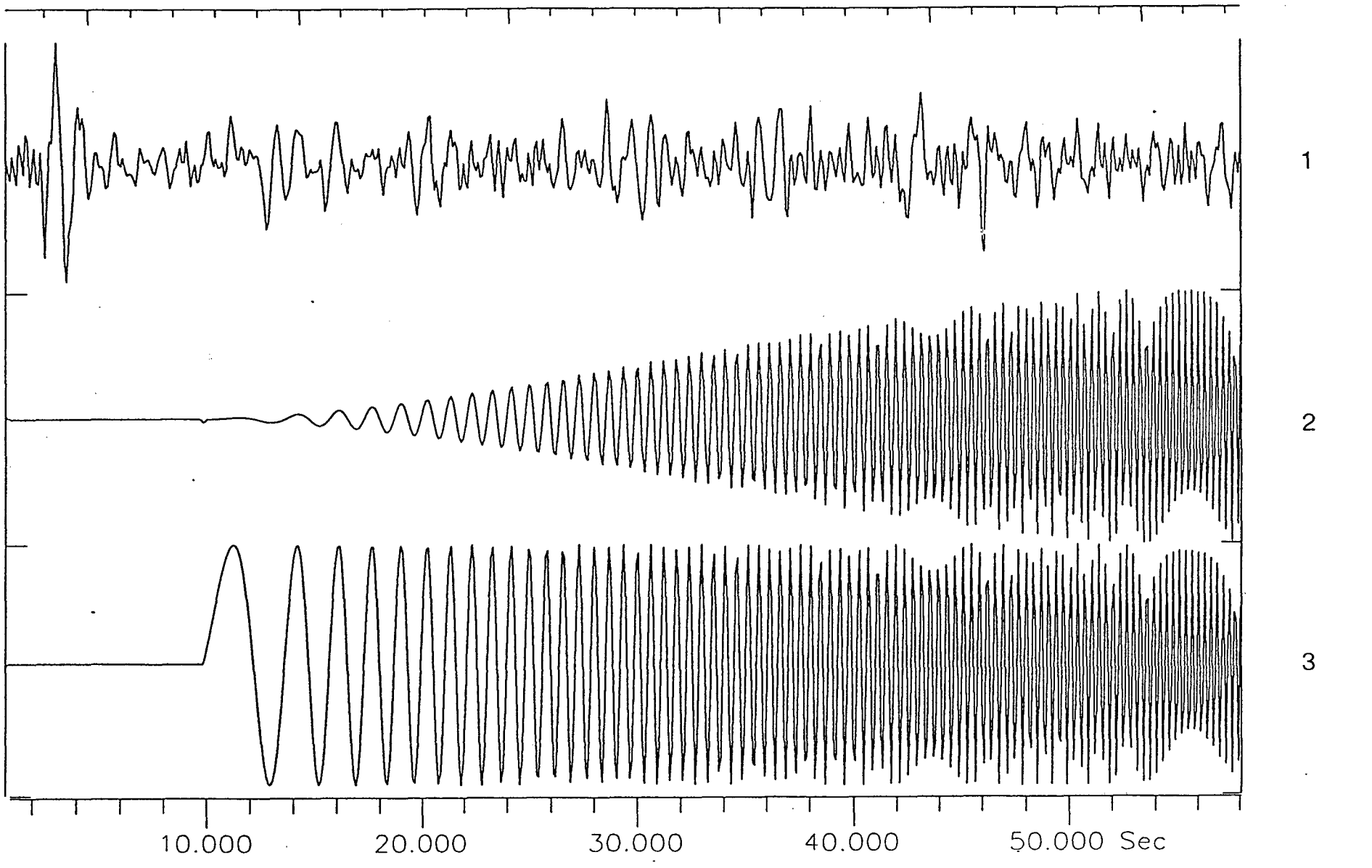

1
2
3
10.000
20.000
30.000
40.000
50.000 Sec
1990—297:15.02.25.711 NRS_sz
Lf= 0 Hf= 5 .
NORESS MODELS: Mixture of Hindu +NZ signals, SNR= 0.130 .

Fig. 9.2.

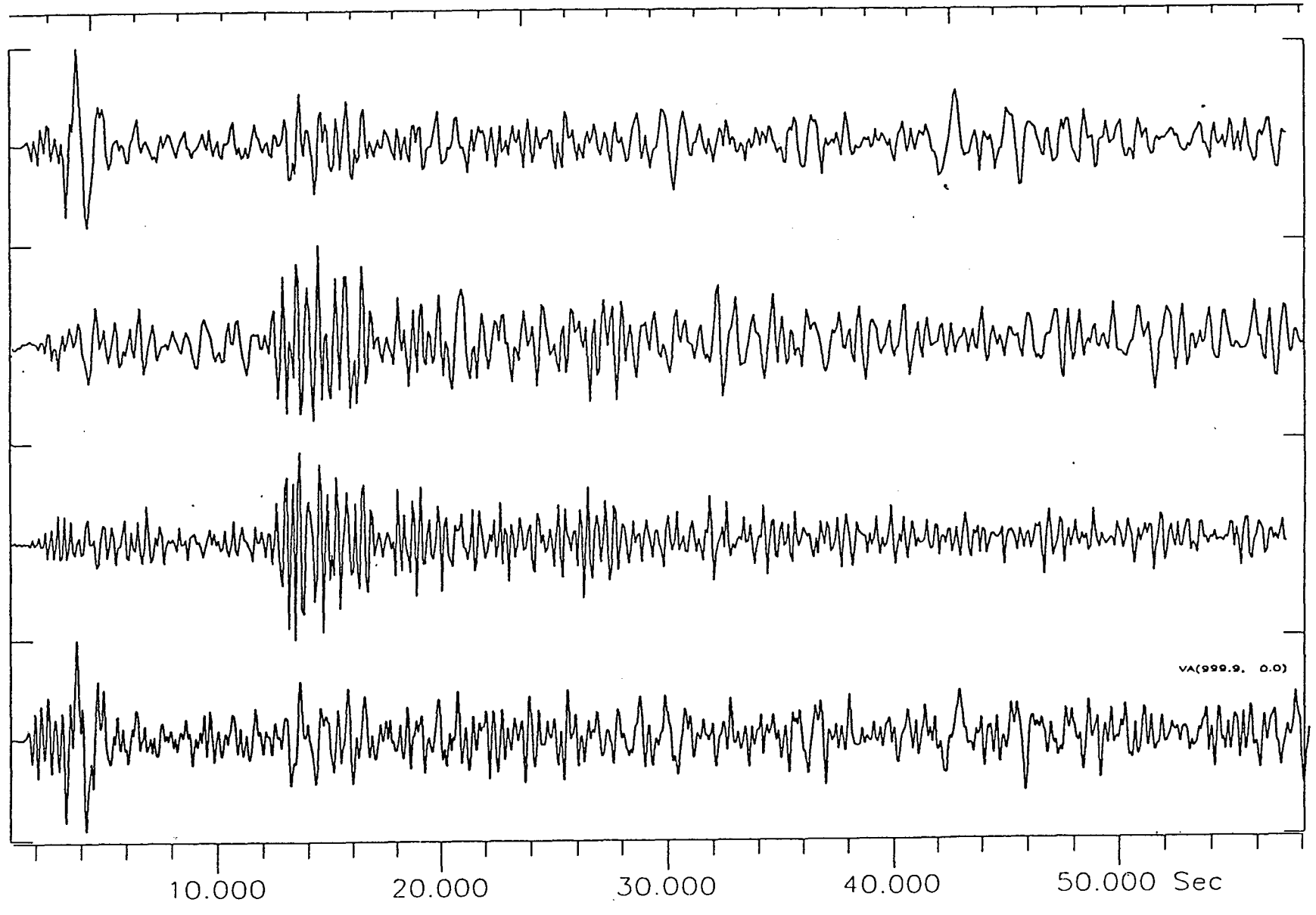

1
2
3
VA(999.9, 0.0)
4
10.000
20.000
30.000
40.000
50.000 Sec
1990-297
Lf= 0 Hf= 5 .
NORESS all, Mixture of Hindu +NZ signals, SNR= 0.30 .

Fig. 9.3.

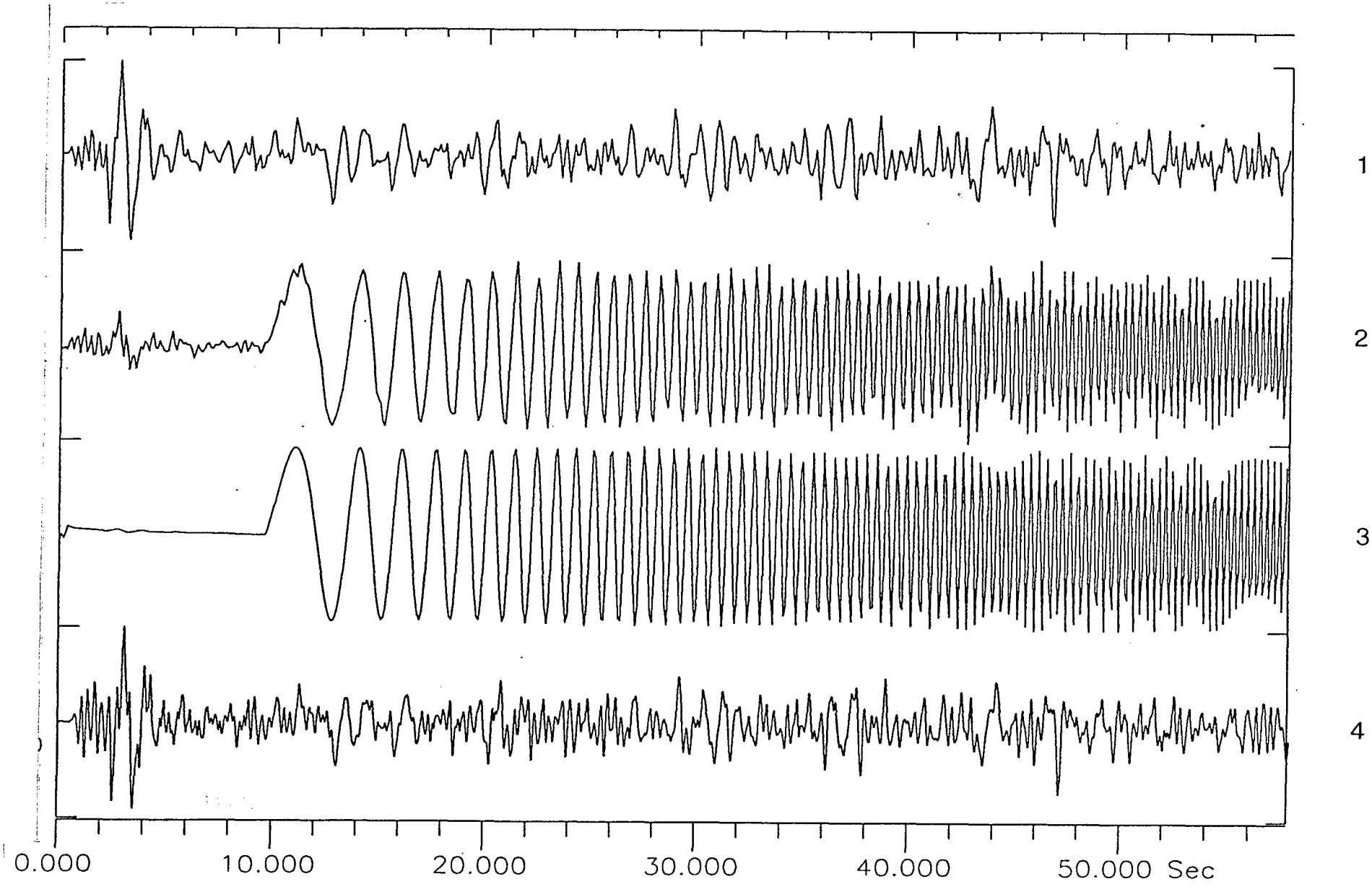

1
2
3
4
0.000
10.000
20.000
30.000
40.000
50.000 Sec
1990−297
ARMA deg:P= 0 Q= 12 Lf= 0 Hf= 5 Reg= 0.010 Ad.md= −3 .
NORESS MODELS: Mixture of Hindu and NZ signals, SNR= 0.13. .

Fig. 9.4.

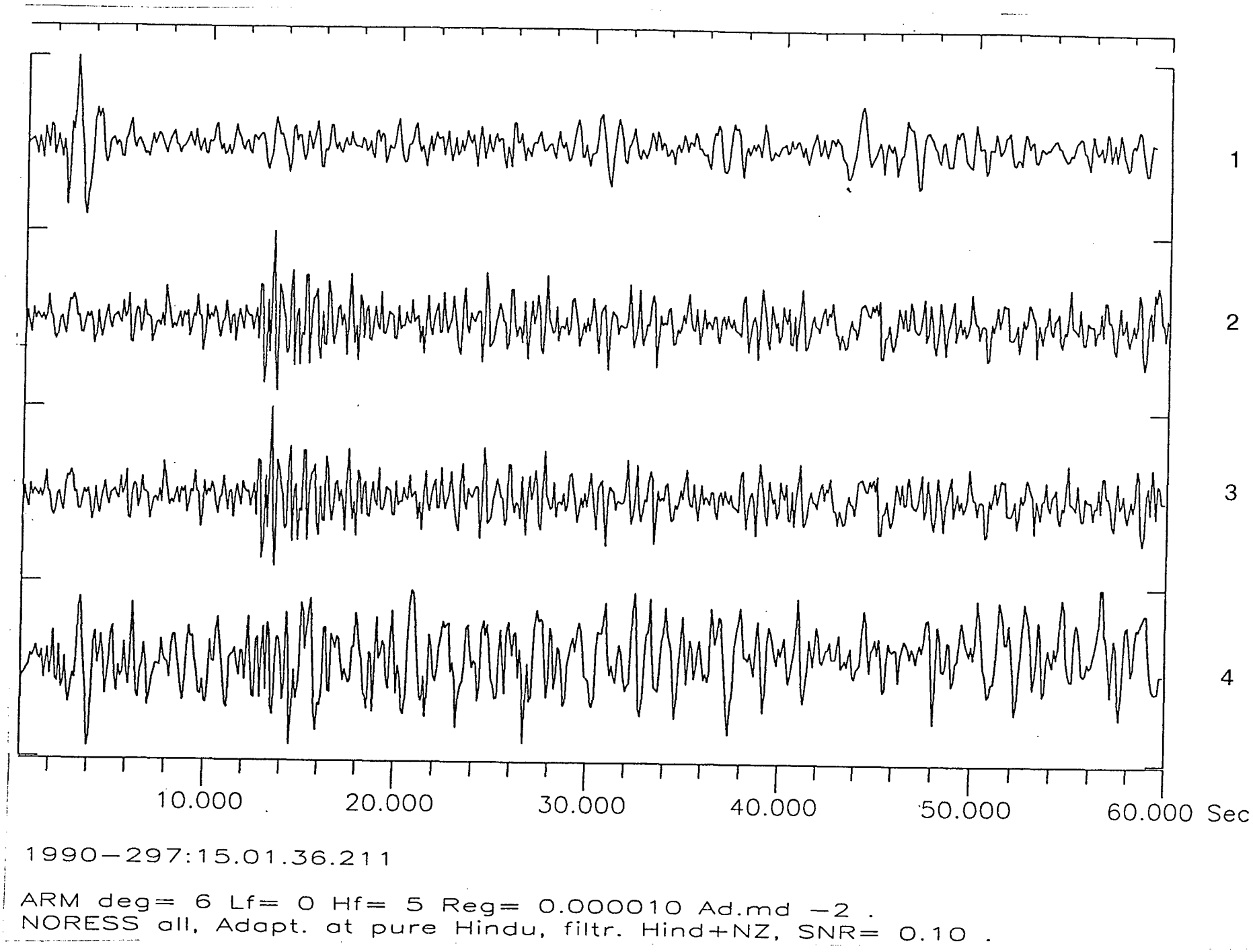

1
2
3
4
10.000
20.000
30.000
40.000
50.000
60.000 Sec
1990–297:15.01.36.211
ARM deg= 6 Lf= 0 Hf= 5 Reg= 0.000010 Ad.md –2 .
NORESS all, Adapt. at pure Hindu, filtr. Hind+NZ, SNR= 0.10 .

Fig. 9.5.

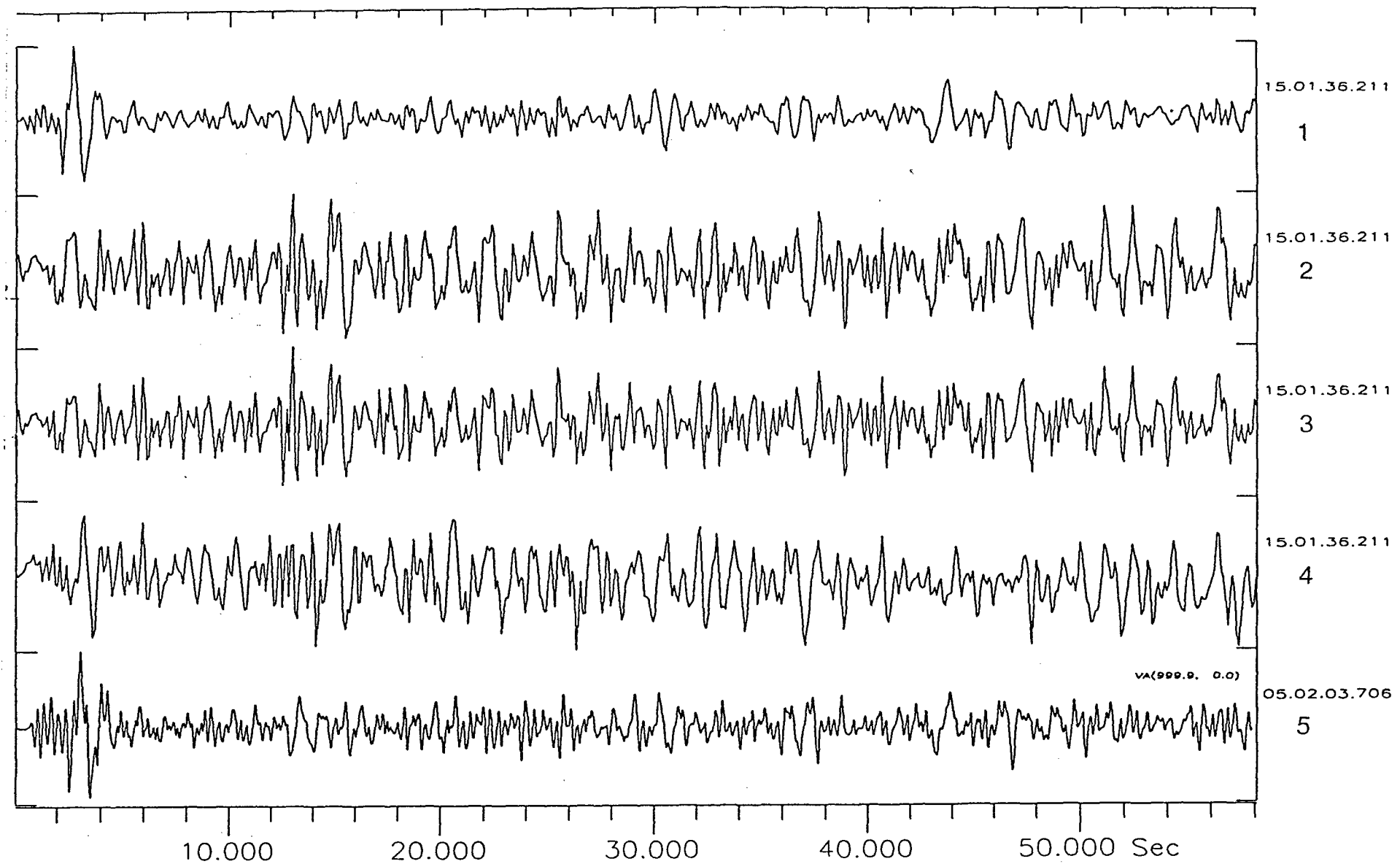

15.01.36.211
1
15.01.36.211
2
15.01.36.211
3
15.01.36.211
4
VA(999.9, 0.0)
05.02.03.706
5
10.000
20.000
30.000
40.000
50.000 Sec
1990–297
A–M mod. deg= 12 Lf= 0 Hf= 5 Reg= 0.001 Ad.md= –3 .
NORESS MODELS: Mixture of Hindu and NZ signals, SNR= 0.10 .

Fig. 9.6.

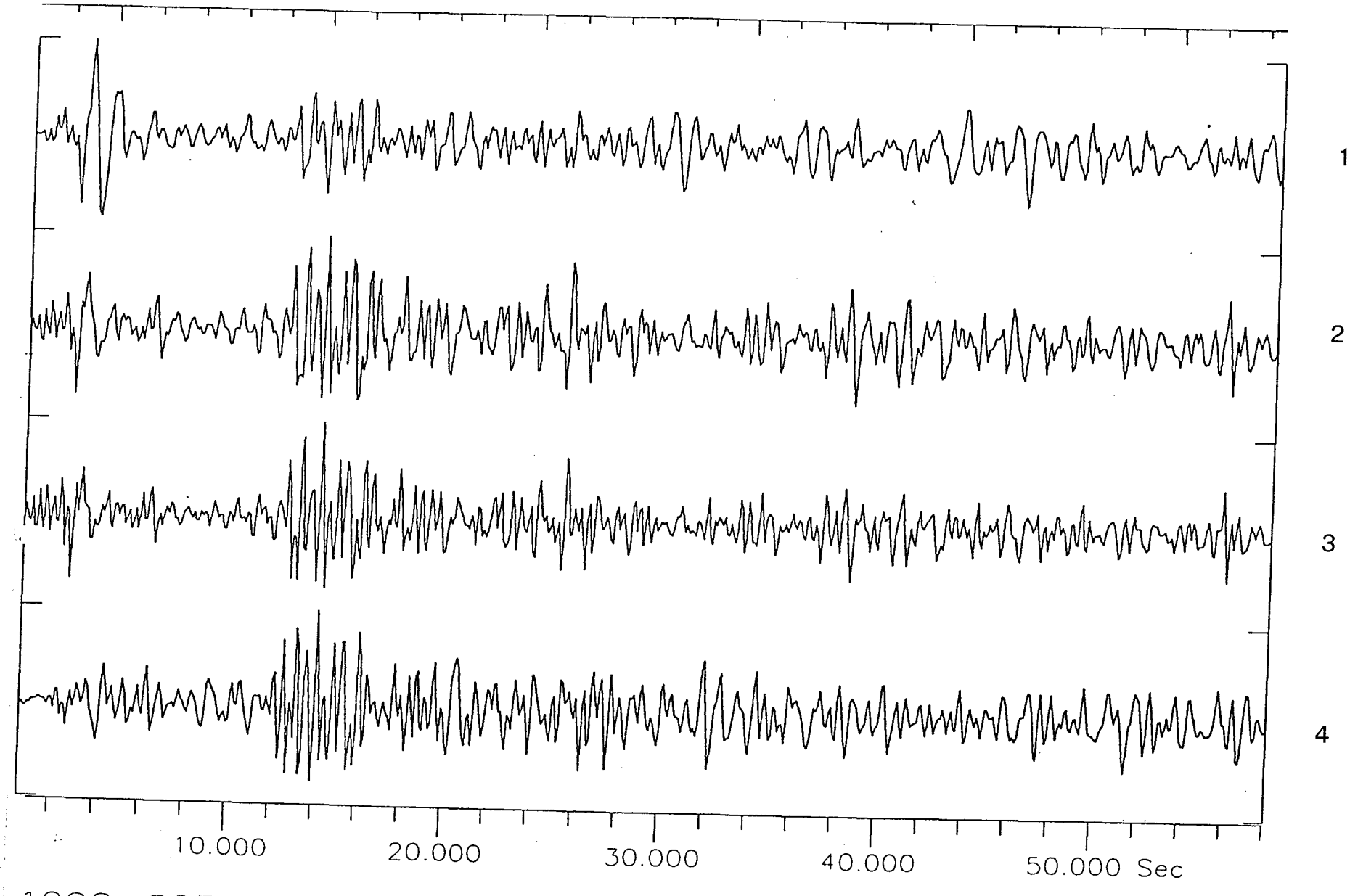

1
2
3
4
10.000
20.000
30.000
40.000
50.000 Sec
1990−297:15.01.36.211
A−M mod. deg= 6 Lf= 0.500 Hf= 5 Reg= 0.010 Ad.md= −2 .
NORESS Mixture of Hindu and NZ signals, SNR= 0.30 .

Fig. 9.7.

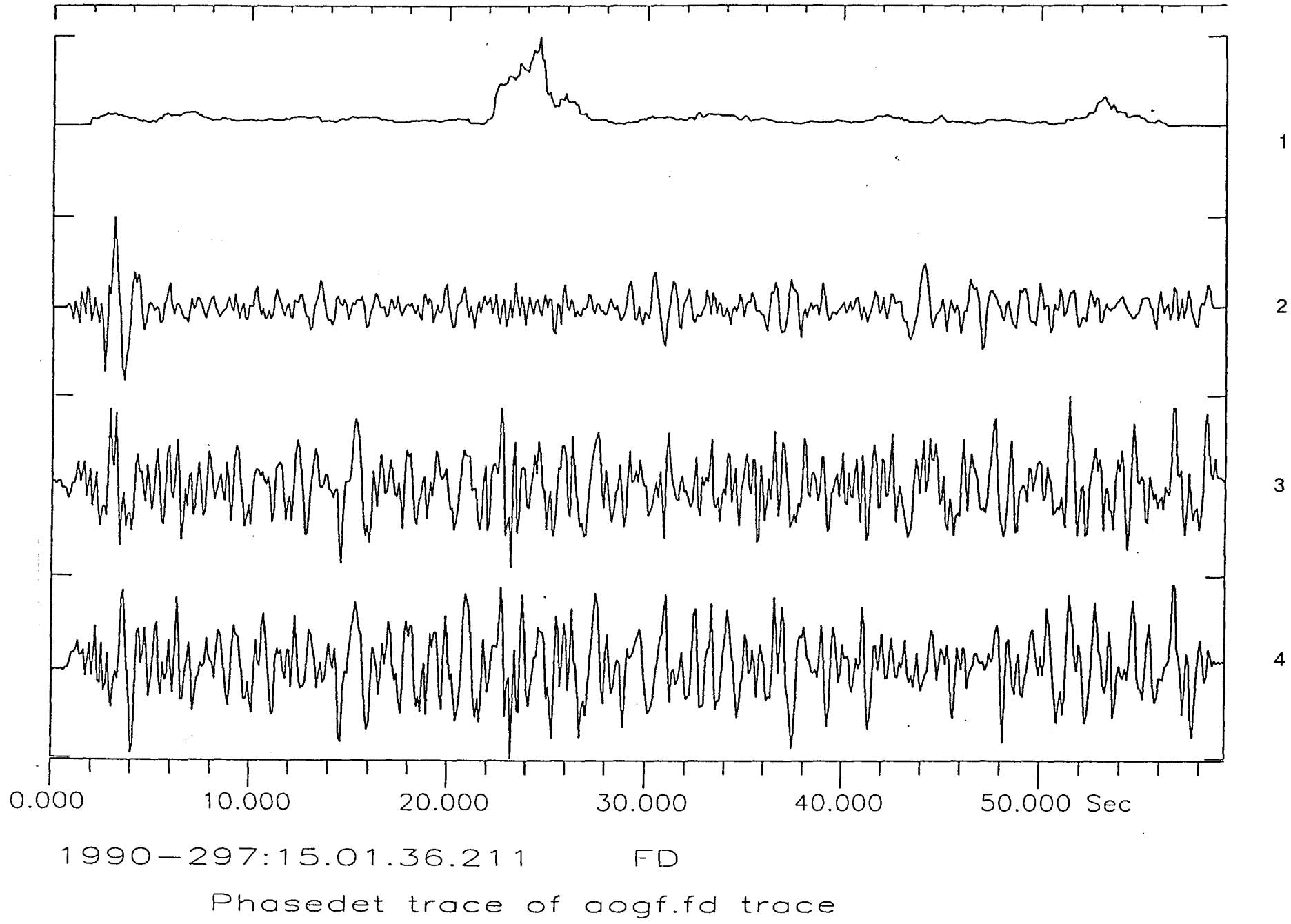

1
2
3
4
0.000
10.000
20.000
30.000
40.000
50.000 Sec
1990—297:15.01.36.211 FD
Phasedet trace of aogf.fd trace

Fig. 9.8.

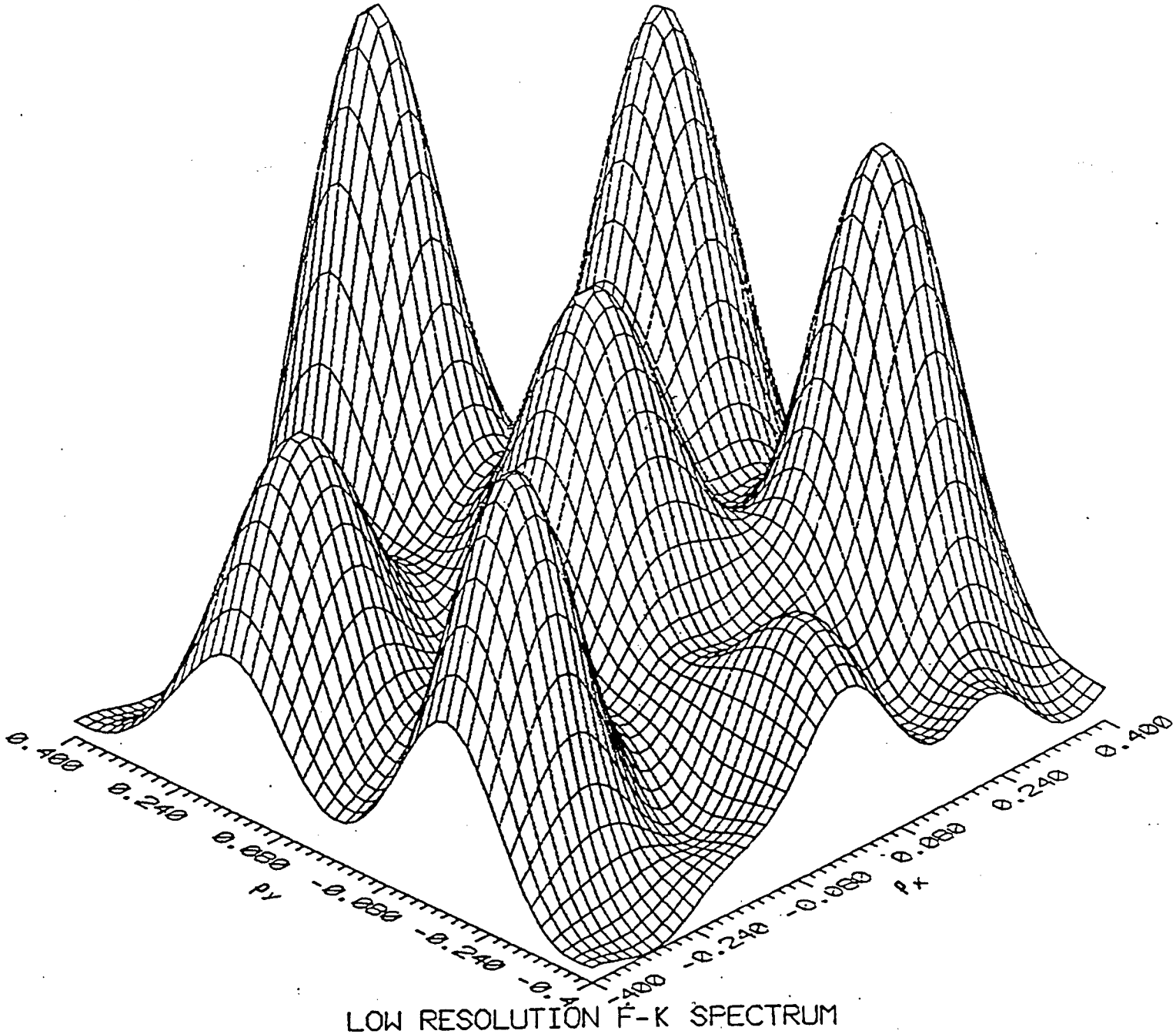

0.400
0.240
0.000
-0.000
-0.240
-0.400
py
-0.400
-0.240
-0.000
0.000
0.240
0.400
px
LOW RESOLUTION F-K SPECTRUM

Fig. 9.9.

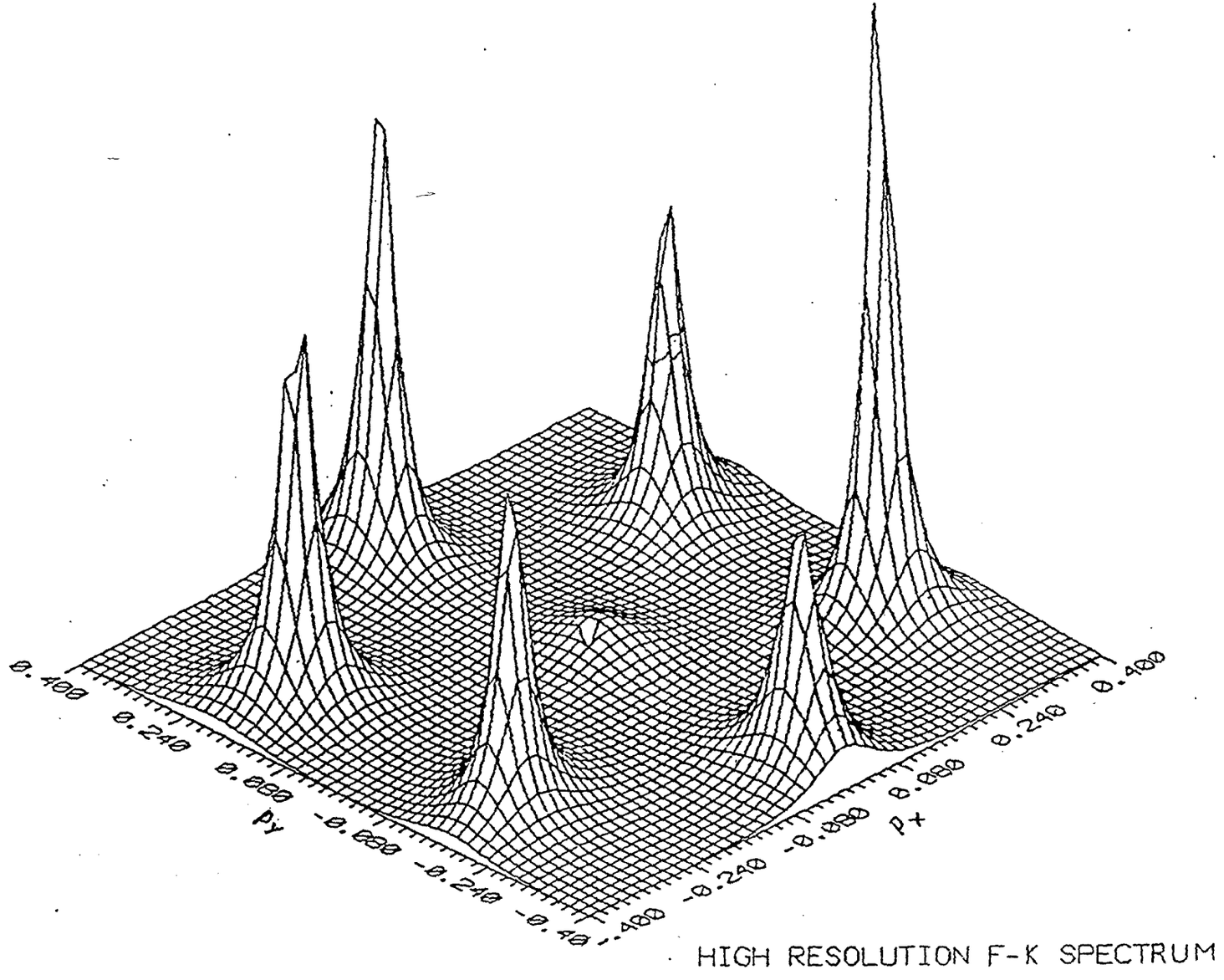
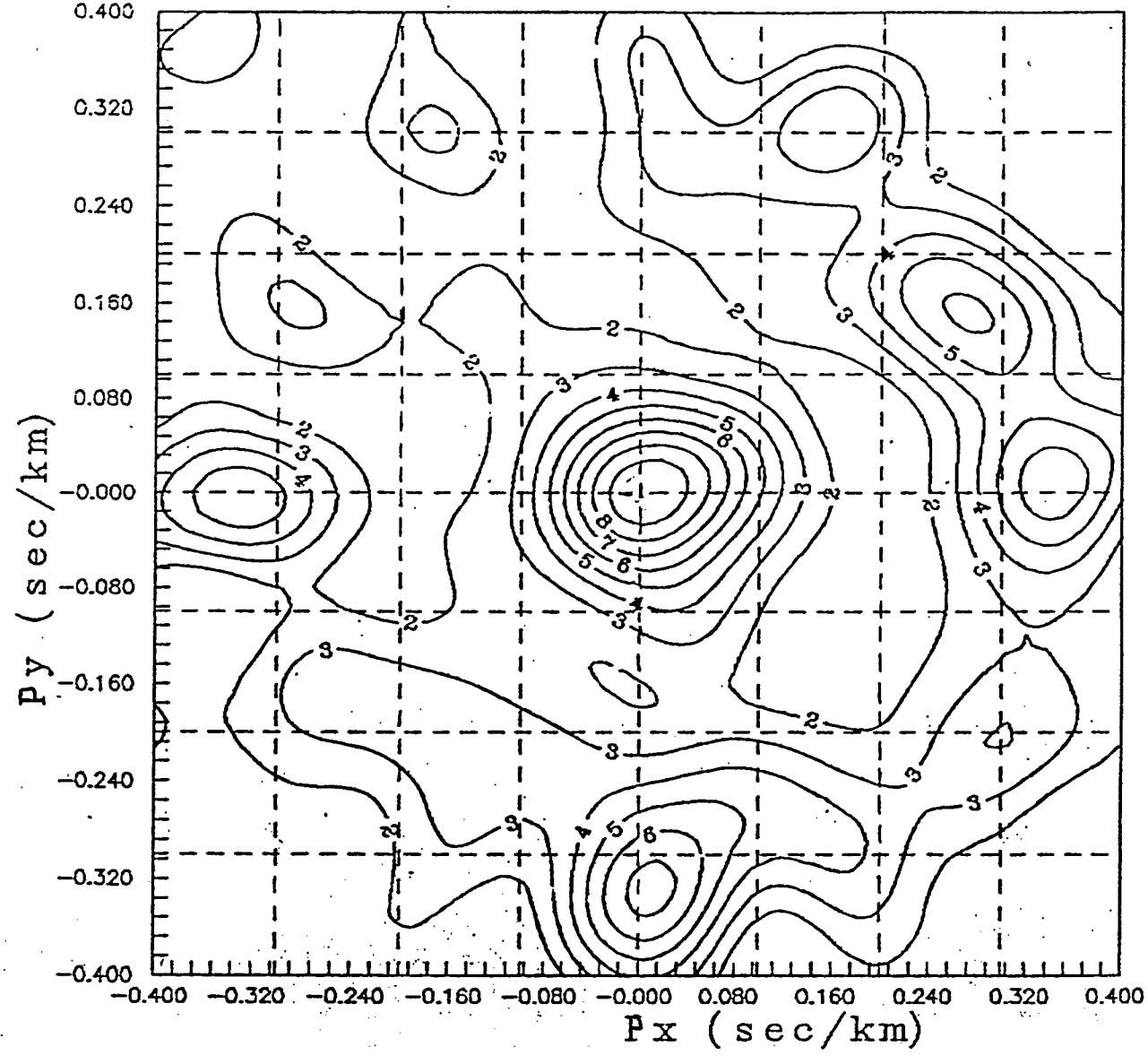

0.400
0.240
0.000
-0.000
-0.240
-0.400
Py
0.400
0.240
0.000
-0.000
-0.240
-0.400
Px
HIGH RESOLUTION F-K SPECTRUM

Fig. 9.10.

LOW RESOLUTION F-K SPECTRUM



freq=3 hz

Fig. 9.11

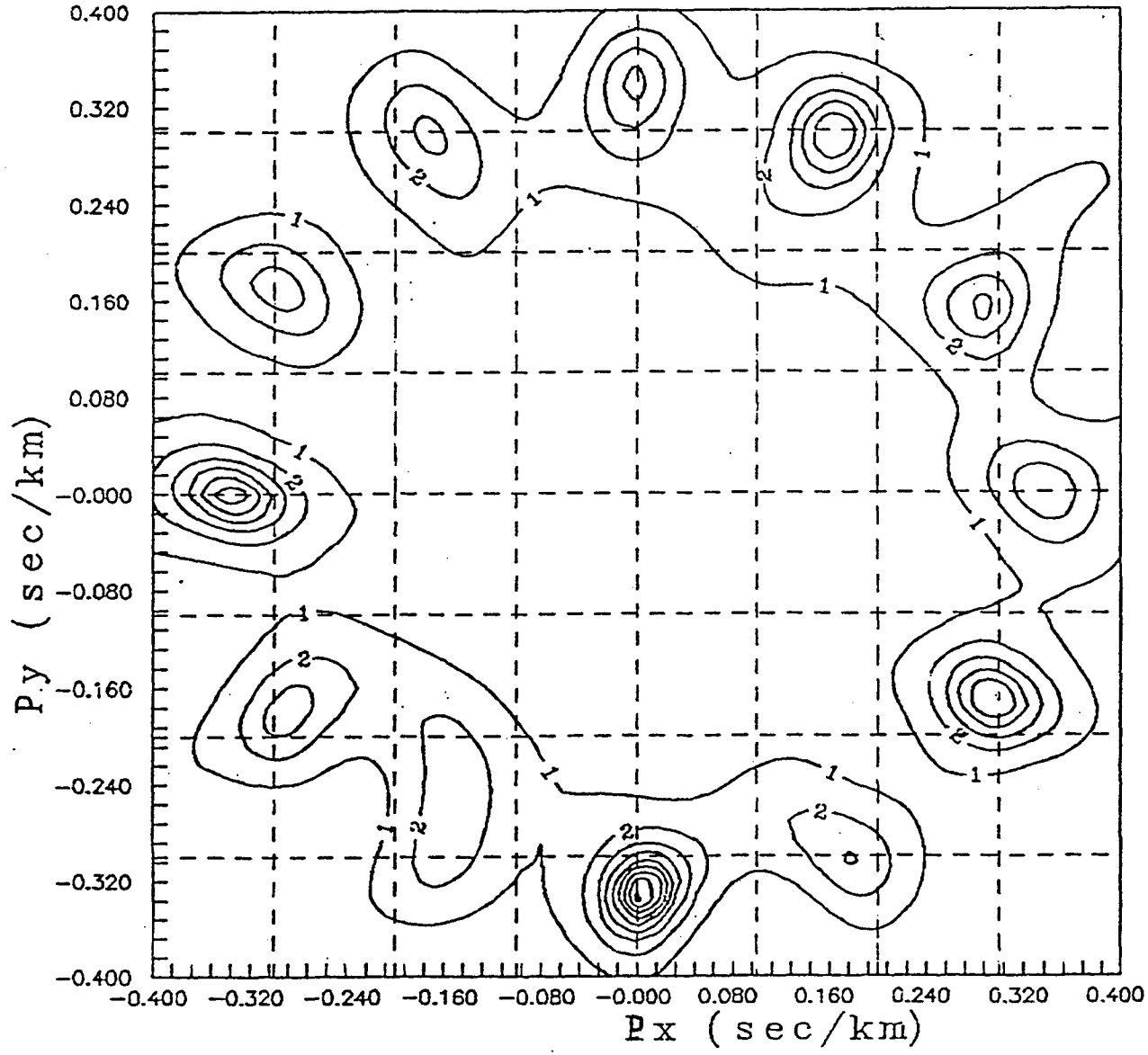

HIGH RESOLUTION F-K SPECTRUM
freq=3 hz
Py (sec/km)
Px (sec/km)
0.400
0.320
0.240
0.160
0.080
-0.000
-0.080
-0.160
-0.240
-0.320
-0.400
-0.400 -0.320 -0.240 -0.160 -0.080 -0.000 0.080 0.160 0.240 0.320 0.400

Fig. 9.12.

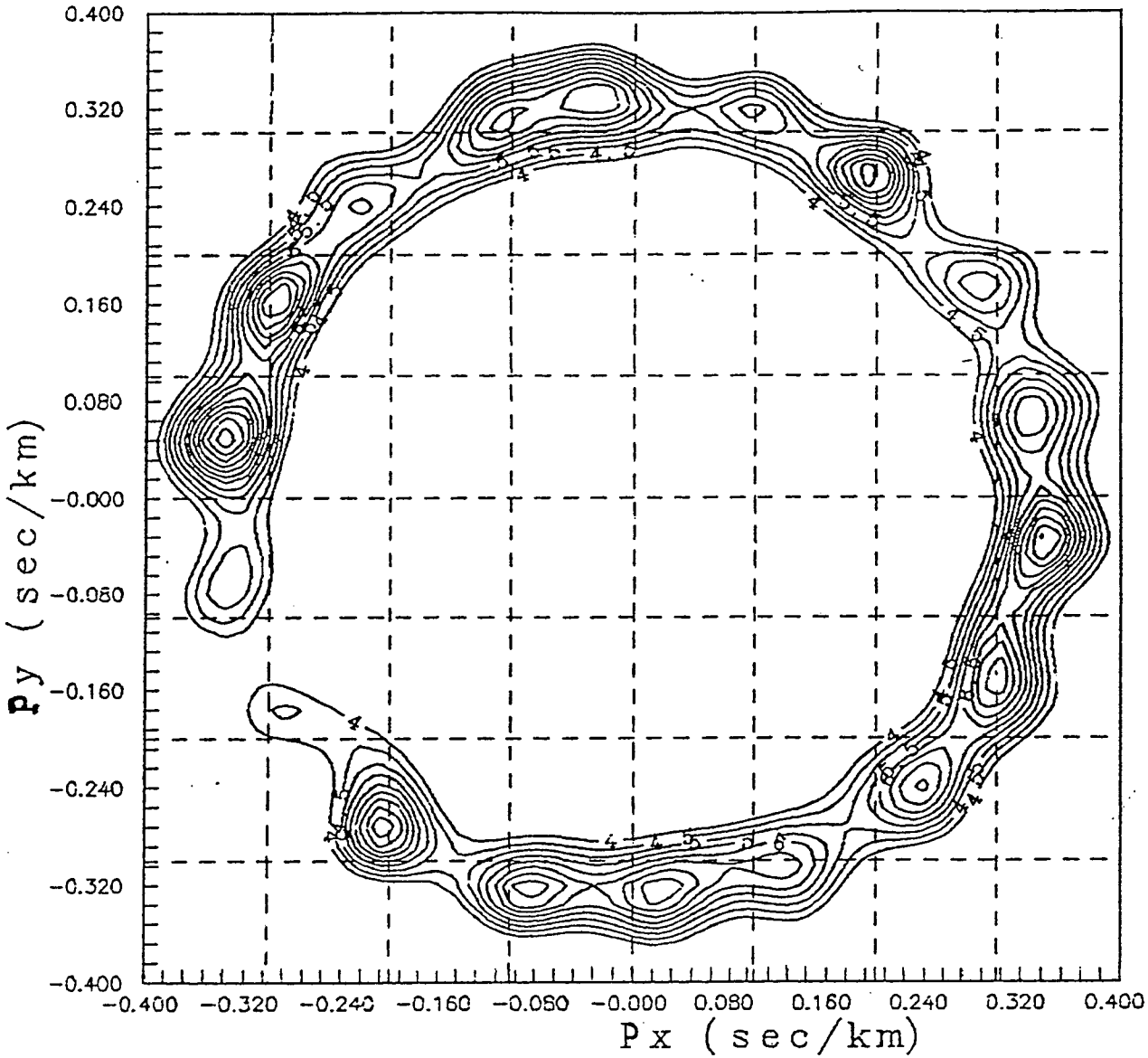

HIGH RESOLUTION F-K SPECTRUM 3hz
0.400
0.320
0.240
0.160
0.080
-0.000
-0.080
-0.160
-0.240
-0.320
-0.400
Py (sec/km)
-0.400 -0.320 -0.240 -0.160 -0.080 -0.000 0.080 0.160 0.240 0.320 0.400
Px (sec/km)

Fig. 9.13.

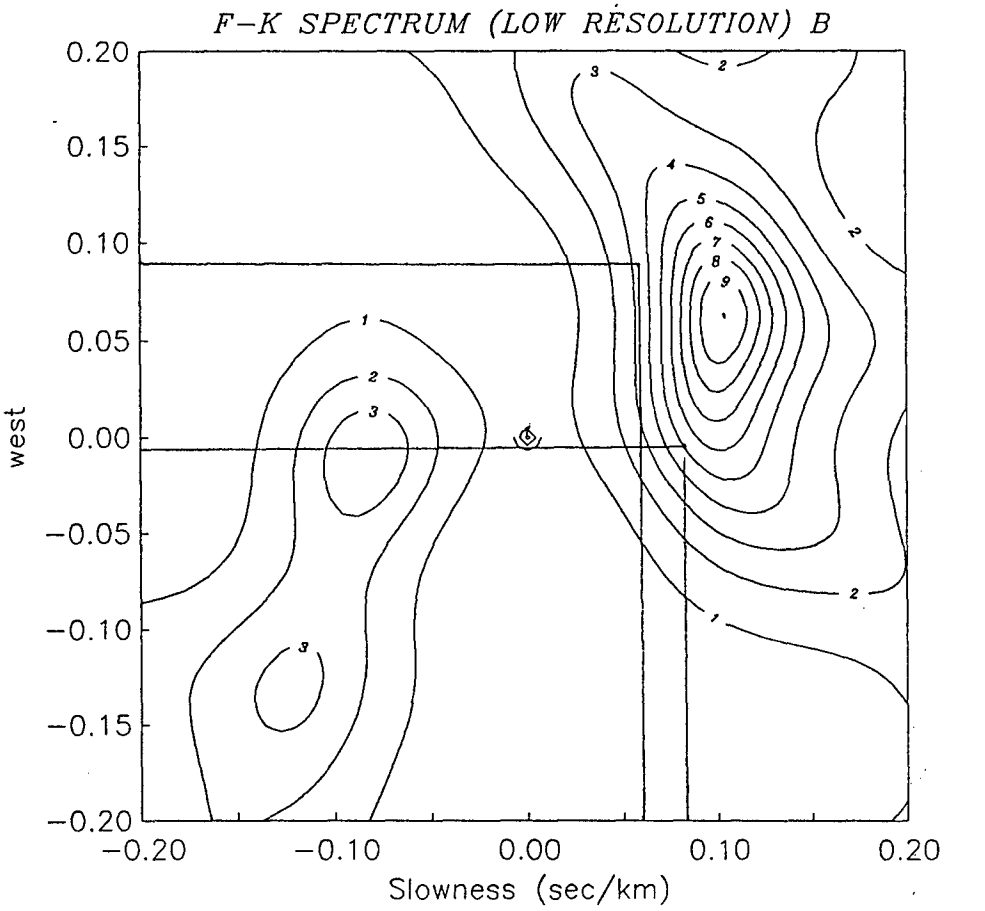

F–K SPECTRUM (LOW RESOLUTION) B
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ,90,298,297,T=2−15,F=3.,P=20,r=.05,snr=1,filt

Fig. 9.14.

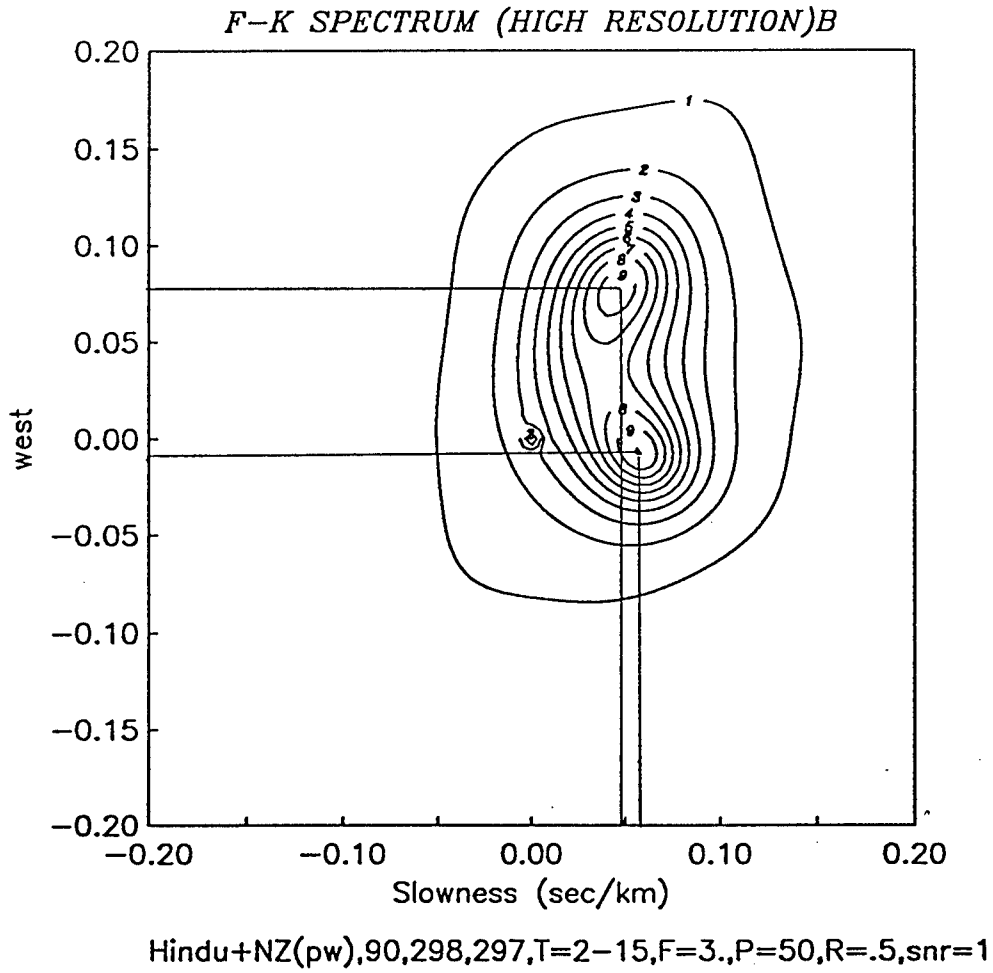

F–K SPECTRUM (HIGH RESOLUTION)B
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ(pw),90,298,297,T=2−15,F=3.,P=50,R=.5,snr=1

Fig. 9.15.

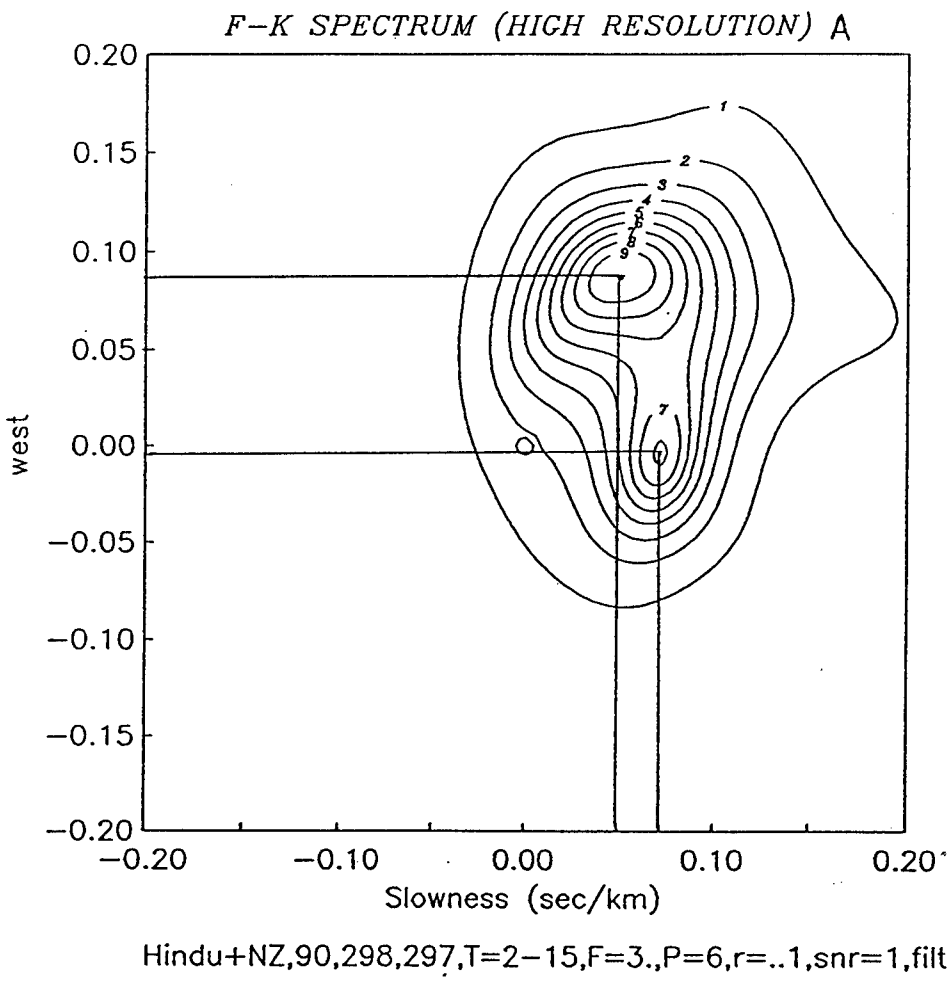

F–K SPECTRUM (HIGH RESOLUTION) A
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ,90,298,297,T=2−15,F=3.,P=6,r=..1,snr=1,filt

Fig. 9.16.

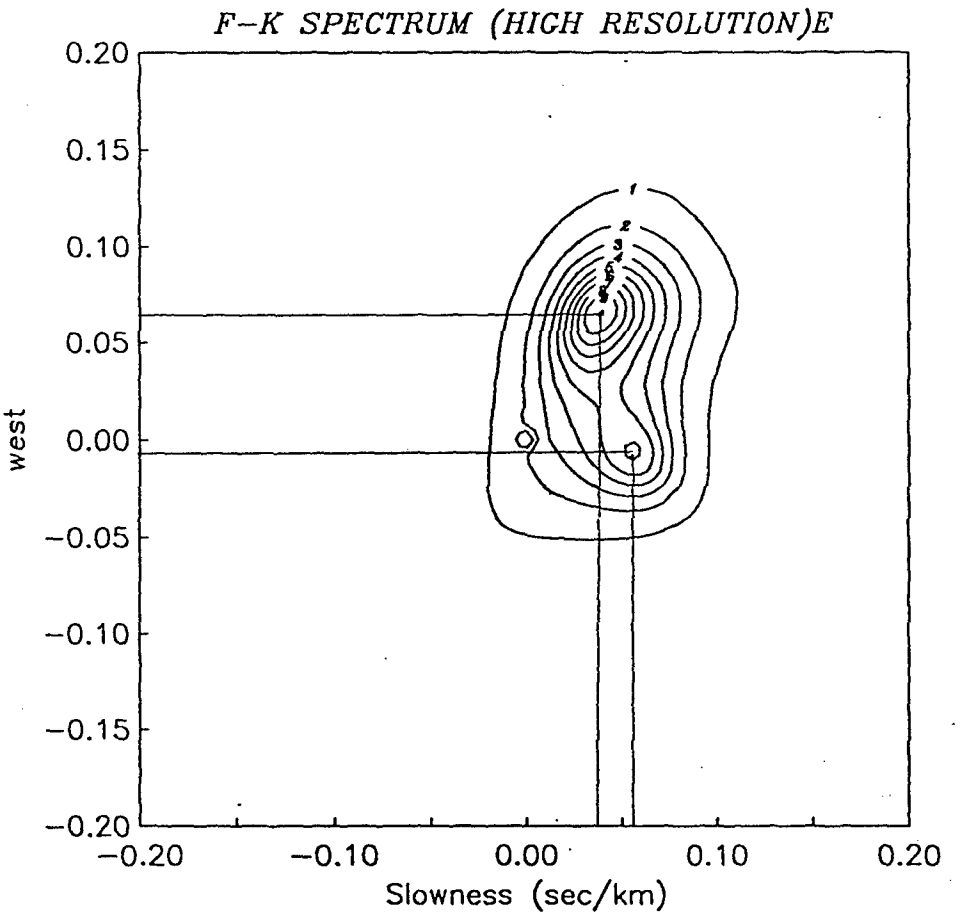

F–K SPECTRUM (HIGH RESOLUTION)E
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ,298,297,T=2−15,F=3.,P=30,R=.1,snr=1

Fig. 9.17.

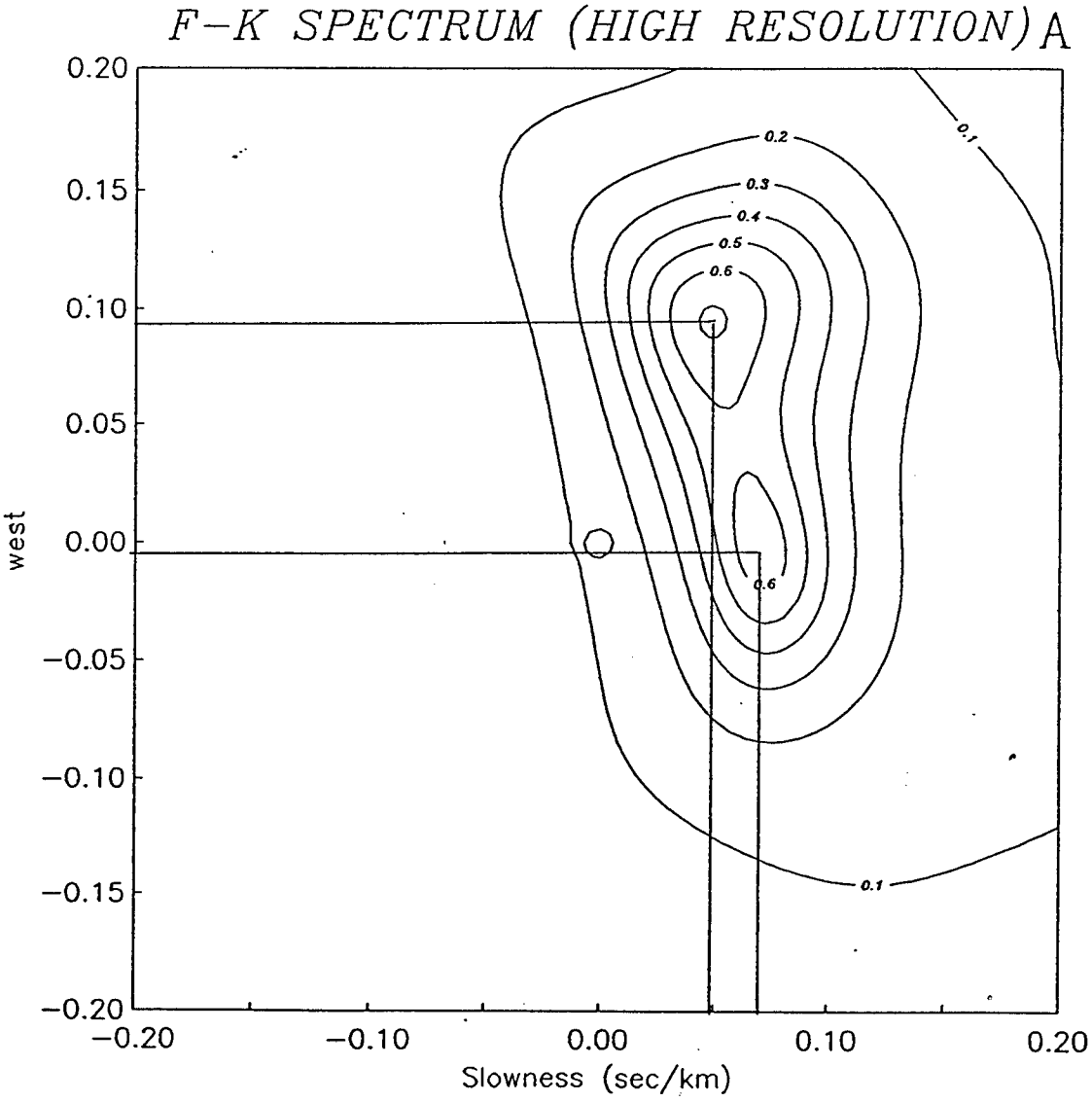

F−K SPECTRUM (HIGH RESOLUTION) A
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
0.2
0.3
0.4
0.5
0.6
0.1
0.6
0.1
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ,90,298,297,T=6−50,F=2.5,P=6,R=.005,snr=.3,filt

Fig. 9.18.

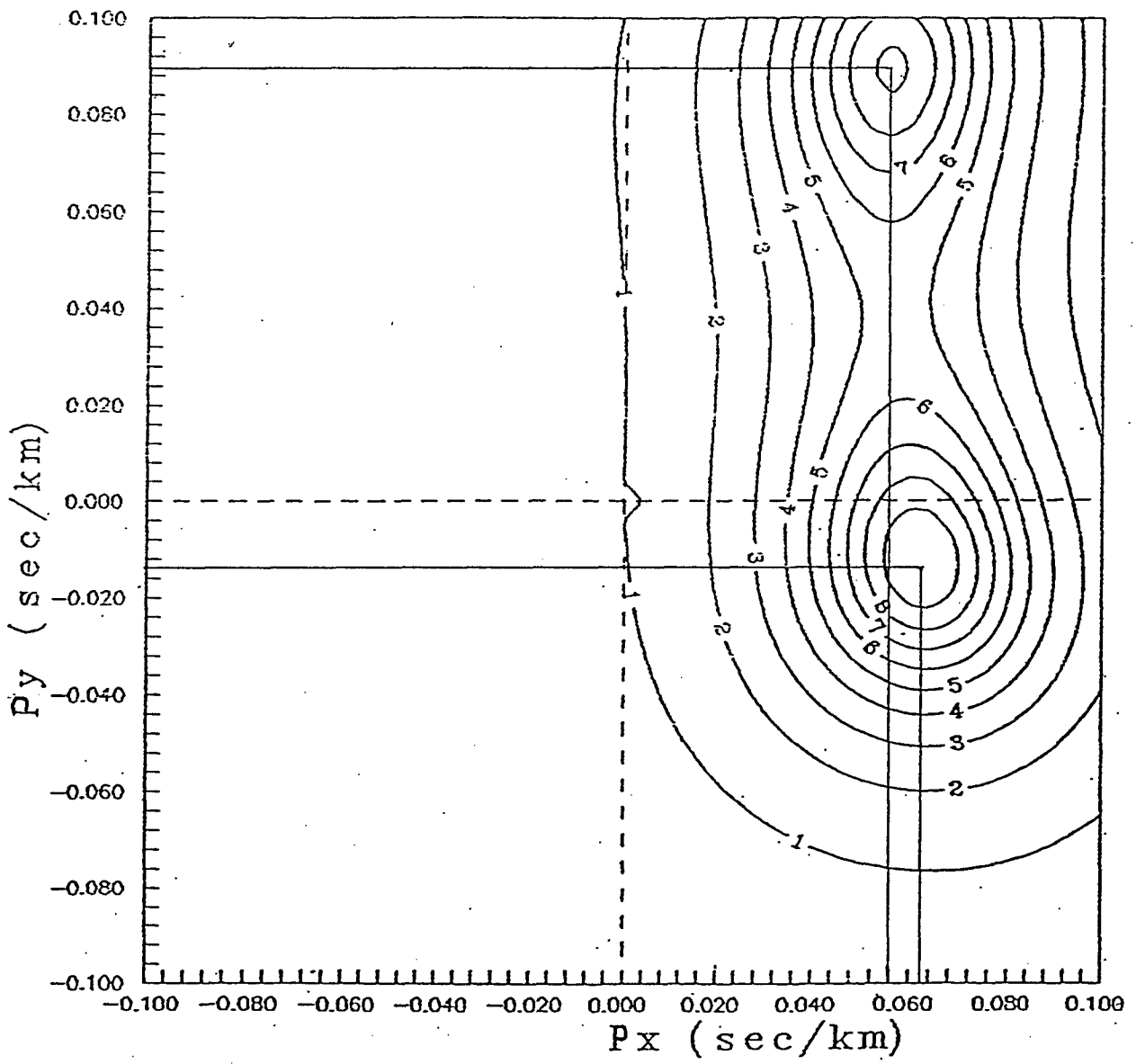

F-K SPECTRUM 'M' (nz+h1)3.2-1.5.6,s/n=0.3
0.100
0.080
0.060
0.040
0.020
0.000
-0.020
-0.040
-0.060
-0.080
-0.100
Py (sec/km)
-0.100 -0.080 -0.060 -0.040 -0.020 0.000 0.020 0.040 0.060 0.080 0.100
Px (sec/km)

Fig. 9.19.

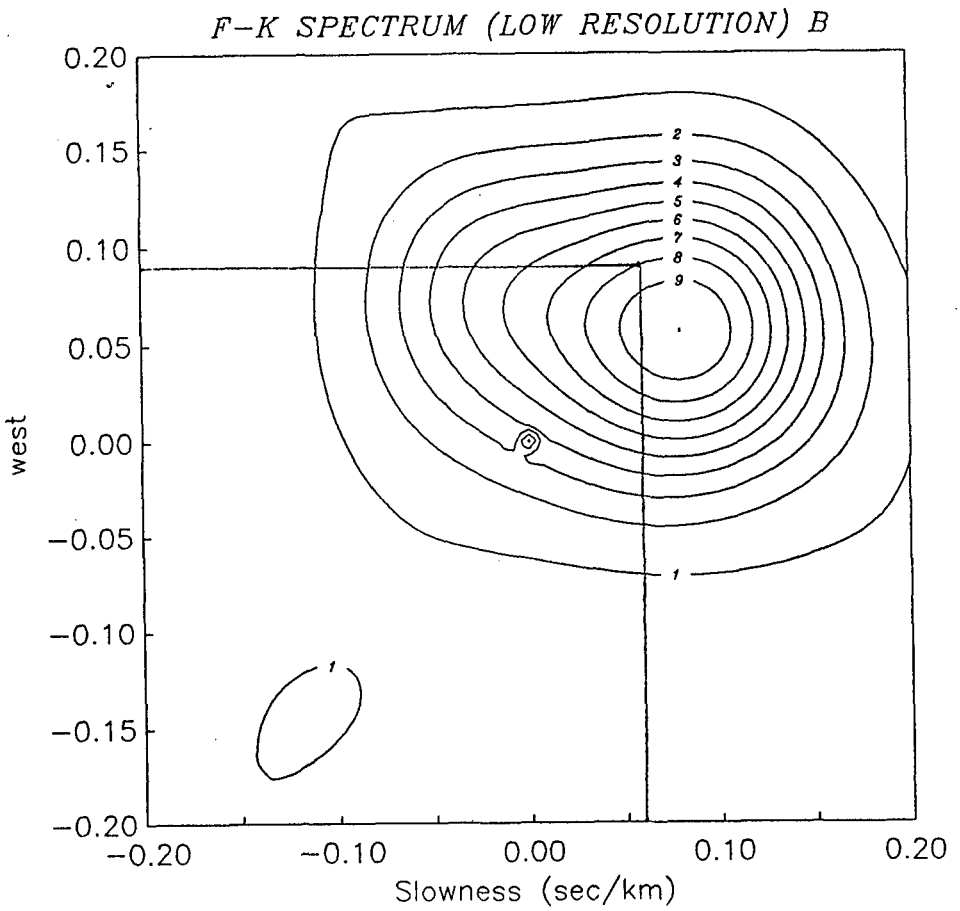

F–K SPECTRUM (LOW RESOLUTION) B
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
2
3
4
5
6
7
8
9
1
1
Hindu+NZ−NZres, T=2−15, F=3, P=20, R=.05,
snr=1

Fig. 9.20.

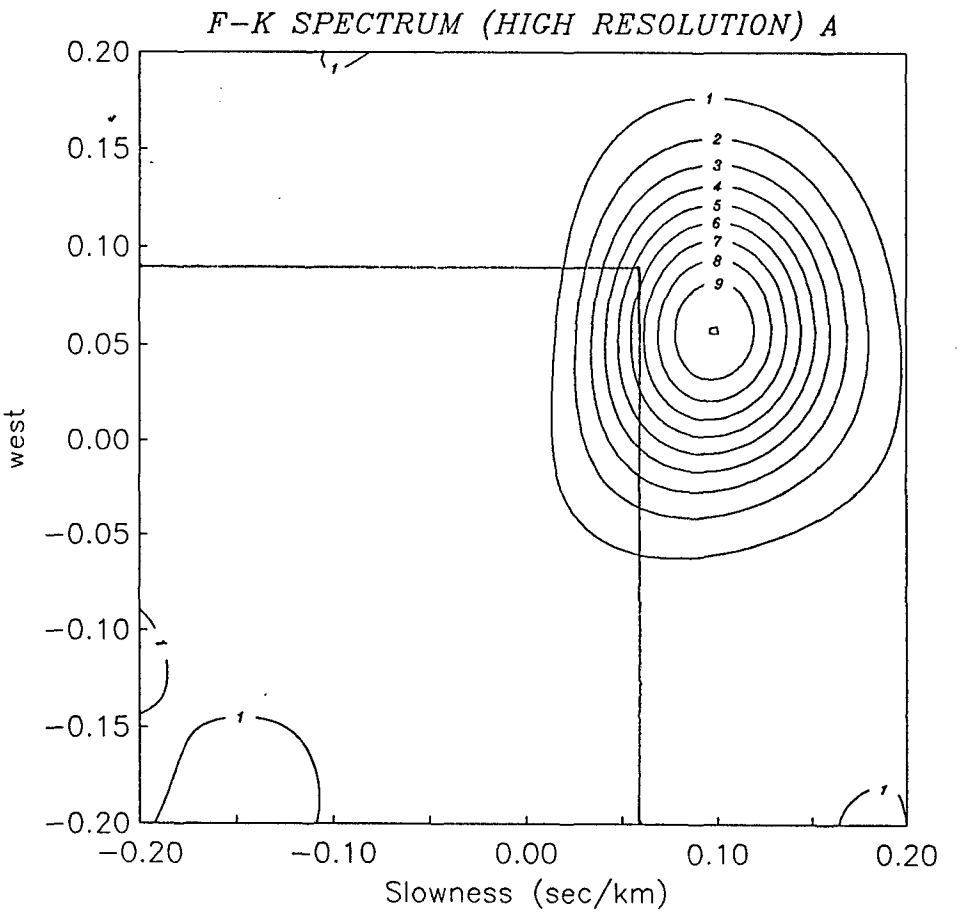

F–K SPECTRUM (HIGH RESOLUTION) A
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
west
−0.20
−0.10
0.00
0.10
0.20
Slowness (sec/km)
Hindu+NZ−NZres, T=2−15, F=3, AP=6, R=.05,
snr=1

Fig. 9.21.

10. Application of adaptive group filtering for enhancing surface wave signals in array records

10.1. Experimental study of low frequency noise suppression using records from small aperture Geyokcha array (Turkmenia)

This section presents a description of experiments on processing of synthetic and real data recorded by the Geyokcha seismic array located In the Ashgabad Region in Turkmenia.

One of the priority trends in applications of statistic adaptive algorithms for processing seismic array data lies in the solution of problems of detecting and measuring surface waves originating from weak earthquakes and explosions. In many cases, a solution to this problem is complicated by the fact that weak low frequency surface waves are registered against the background of intensive diffuse and correlated noises of different nature whose statistical properties vary temporally and spatially in an unpredictable manner and whose frequency spectrum overlaps with those of surface waves. Under these conditions, detecting and measuring surface waves often proves to be unsolvable problem in the framework of traditional methods of seismological data processing. One of the approaches to solving this problem is the application of adaptive group filtering. The parameters of an adaptive filter vary with the properties of input noise, adjusting themselves automatically to optimum signal extraction. An experimental program for real and synthetic data processing under the present project has been aimed at solving the following problems:

(1) Integrated testing and assessment of the efficiency of the main programs for adaptive group filtering and the SNDA package as a whole in the real data analysis regime.

(2) Estimation of parameters of real signals and noise recorded by the Geyokcha array. The assessment of possibilities and quality of detecting and extracting weak surface waves generated by teleseismic events against the background of intensive noises.

(3) Assessment of efficiency of the real data processing using programs written in the Job Control Language of the SNDA package (JCL SNDA).

Under this project, a number of programs (processing graphs) in the JCL SNDA were developed which were used to process different seismograms recorded by the Geyokcha seismic array. Examples of two JCL

programs with comments are presented below in this Section. It is noteworthy that we had a limited amount of real seismograms at our disposal. For this reason in the surface waves extraction experiments we used real records of seismic noise and the synthetic seismograms of weak surface waves added to these noise. The parameters of these surface waves matched regional events in the region of the Lob Nor testing site in Northwestern China. In the data processing scripts, in addition to all the main SNDA Stack commands and the general purpose JCL commands, the following procedures for the analysis of seismic data (SA procedures) were used:

- frequency filtering:	fitterC, filterS
- power spectrum analysis:	powspec
- autoregression- moving average multi dimensional modeling of seismic noise	marmamo
- spatial spectrum analysis:	spsp1, tk1
- kinematics forward modeling:	estimtt
- design of frequency responses for the optimal group filter:	armagrf
- adaptive group filtering:	fpsfa

Analysis of the results of Geyokcha seismic array data treatment indicates the capacity for work of the adaptive group filtering algorithms, the high efficiency of enhancing weak signals, the simplicity and reliability of implementation of various scripts for treating large volumes of data with the help of well developed JCL SNDA.

The small-aperture Geyokcha seismic array is situated in Turkmenia not far from Ashgabad. It includes 12 wide-band three-component seismometers STS-2, composing a subarray with the configuration shown in Fig. 10.1. The coordinates of the central station ORGH are: Lat. 37.92933, Long. 58.11250, Elev. 662.9 m. The array is situated in region of thick sedimentary rocks.

Fig. 10.2 a,b,c shows three vertical component seismograms with one hour duration. The seismograms were recorded on 24.04.94 by the Geyokcha subarray consisting 12 wide band seismometers. Some weak local events are recorded on seismograms Fig. 10.2a and 10.2b. Fig. 10.3 a,b,c presents 200 sec. intervals of above mentioned seismograms, and Fig. 10.4 a,b,c presents the power spectra of above 200 sec. records. Every latter Figures contains the spectra of 3 different channels and the spectrum of a beam trace (marked as

ORGH) which is merely the sum of the all 12 channels. One can see that in all the seismograms the bulk of energy recorded is accounted for by low frequency oscillations (from 0.05 up to 1.0 Hz) with the high degree of cross correlation. These oscillations are most likely related to storm microseisms.

Fig.10.5 shows one of the F-K analysis results. Analysis was made in the frequency band 0.1 - 0.5 Hz. The F-K analysis results demonstrate that the seismograms are constituted mostly by noise oscillations formed by waves with apparent velocities about 25km/sec - 40km/sec and azimuths about 200 - 240 deg. The noise sensor recordings with the high degree of cross correlation is the characteristic feature of small aperture seismic arrays. This create numerous problems in the obtaining reliable information about seismic events.

Using the SNDA program "estimtt" for kinematics propagation modeling (ray propagation approximation), we estimated the surface wave parameters from shallow source in northwestern part of China. These parameters are: App.Vel. 3.55km/sec, Azimuth =71.6967 deg, Distance: 23.7043 deg, the travel time for LG phase Tlg=12 min. 33.085 sec and for RG phase Trg=14 min. 52.968 sec. Subsequently, we use a simulation program "bwarsim" to develop the surface wave synthetic seismograms from Geyokcha array simulating surface waves arriving to the array with the above mentioned azimuth and apparent velocity. The seismograms consists of two Berlage pulses with onsets on 1750 sec and 2250 sec from the beginning of seismogram. The 12 channels of array synthetic seismogram is shown in Fig. 10.6. The specified frequency range of the synthetic seismogram (0.15 Hz - 0.25 Hz) was chosen to be inside the frequency range of the time-correlated seismic noise and synthetic signal power was set 100 times less that average power of the real noise. This synthetic seismograms were added to real noise seismograms shown in Fig. 10.2 a,b,c. The above process of mixture data simulation has been accomplished using a script 'albmsde.scr', i.e. processing graph written in the SNDA JCL. The text of this script with comments is given below (Script 1).

The obtained seismograms which contain the mixture of the real seismic noise registered by Geyokcha array and the weak modeled surface wave signals were processed by different array data processing algorithms: beam forming, rejecting filtering, adaptive group filtering, whitening filtering. These methods is discussed in the details in previous sections of the report. In these experiments the main idea of adaptive seismograms processing is reduced to design of the optimal (in the Wiener sense) group filter. The information needed to design the filter frequency response function is

accumulated by the estimation of noise matrix power spectral density on the record interval which does not contain any signals (the filter adaptation). Based on this estimation of the seismic noise spectral characteristics we perform both the calculation of filter frequency response and the optimal group filtering of seismogram. This optimal group filter (UGRF_FD) provides the well noise suppression in wide ranges of frequencies and apparent velocities with the best retention of a signal shape. The spatial rejecting group filtering (REJ_FD) performs the best suppression of interfering waves in the given narrow slowness range. The whitening group filter (WGRF_FD) provides the efficient destroying of correlated noise, resulting in the greater probability of weak signal detection. This filtering may be accompanied by some output signal shape distortion.

The processing graph aimed to surface waves enhancement includes all above mentioned adaptive algorithms. It was realized by a script "alsmprps.scr". The text of this script with comments is given below (Script 2). Some results of processing by this script the two from above mentioned seismograms are shown in Fig. 10.7 a,b,c. Two vertical lines indicate onset times of the two simulated surface waves. These examples illustrate both a high rate suppression of correlated noise in the small aperture array data and high efficiency of the weak signal extraction.

10.2 Enhancement of surface waves in records from large aperture NORSAR and GRAFENBERG arrays

The large aperture seismic arrays such as NORSAR and GRAFENBERG are equipped with the long period (LP) 3-component seismic instruments recording seismic signals in frequency band 0.001 - 0.5 Hz with sampling frequency 1.0 Hz. This is just the frequency range to register surface waves from teleseismic and even regional events. The wide spread of the LP sensor location where the distance between the sensors are in limits 20 -100 km implies that the assumption on homogeneity of medium beneath the array became unrealistic even for low frequencies. For this reason noise records from LP sensors of these arrays do not exhibit explicit visual coherency (see Fig.8.5). Thus, an investigation of optimal group filter capability to enhance weak surface waves in recording of LP subarrays of large aperture arrays is a challenge for this method potentials, especially if only 5-6 distant sensors are used.

Dr. Anton Dainty provided us with records of teleseismic surface waves from 4 nuclear explosions registered by Long Period Subarrays of NORSAR and GRAFENBERG arrays. The configurations of sensor locations for these subarrays are shown in Fig. 10.13 a,b. These records were treated with conventional beamforming (BF), adaptive optimal group filtering (AOGF) and adaptive whitening group filtering (AWGF) procedures. The data processing was accomplished in the framework of the SNDA with the help of its script facilities. An example of the script for surface waves enhancing is given at the end of this Section (Script 3).

As the teleseismic surface waves (SW) were processed all the array records were treated with low pass filtering with high cut frequency 0.1 Hz with subsequent resampling, providing 0.1 Hz to be the Nykuist frequency of the filtered data. As the azimuths and apparent velocities of the surface waves at this array recordings were known there was no problems with steering the group filters in the correct directions. The signal-to-noise ratio of surface waves registered became high enough after LP filtering to allow visual detecting this waves at the seismograms. This enables us to select intervals at the seismograms before and after SW waveforms, where only noise can be considered as present. These intervals (marked off by the vertical lines in the Figures below) are used for adaptation of the optimal group filters.

The results of data processing are shown in Fig. 10.8-10.12. For every figure there are three plots. At the first (a) plot the array seismograms LP filtered up to frequency 0.1 Hz are shown. The vertical lines define the time interval which is not used for the group filter adaptations. At the second plot (b) the processing results are demonstrated in the same time scale as for the previous plot (a). The trace with the label BEAM_FD is the result of conventional beamforming procedure applied to LP array records in the frequency domain. The trace with the label UNGFR_FD is the result of adaptive optimal group filtering which do not distorts the signal waveform. The trace with the label WGRF_FD is the result of adaptive group filtering which whitening the residual noise but distorts the signal waveform. The bottom trace at the plot (b) is the trace from one of the array sensors. It is drawn for comparison with the group filtering results. At the plot (c) the same traces as for plot (b) are drawn inside the time interval which was not used for group filter adaptation. The large tine scale at this plot allow to analyze signal and noise waveforms and compare signal shapes after BF, AOGF and AWGF procedures.

An analysis of the Fig. 10.8-10.12 allow to assert that for all the array data samples processed the adaptive group filtering provides better results

then conventional beamformig procedure. The SNR of surface waves was enhanced, for some cases the gain appears to be rather significant. Especially strong gain is provided by the adaptive whitening group filtering due to accounting the difference in frequency content between noise and surface wave oscillations. This filter is proved to be useful for preprocessing before detecting weak seismic phases.

The Fig. 10.8 and Fig. 10.9 exhibit results of processing the records from horizontal and vertical LP NORSAR 3-component sensors made for the same teleseismic event. A comparison of the Figures make sure that the SNR gain for vertical component is greater then for horizontal component.

We regard this preliminary results of adaptive group filtering method application for seismic waves extraction from low frequency coherent noise as encouraging for the further investigation of this method applied to LP array data processing

The investigations performed in the real data processing with the set of an adaptive statistical algorithms incorporated in the SNDA package allow to assert the following:

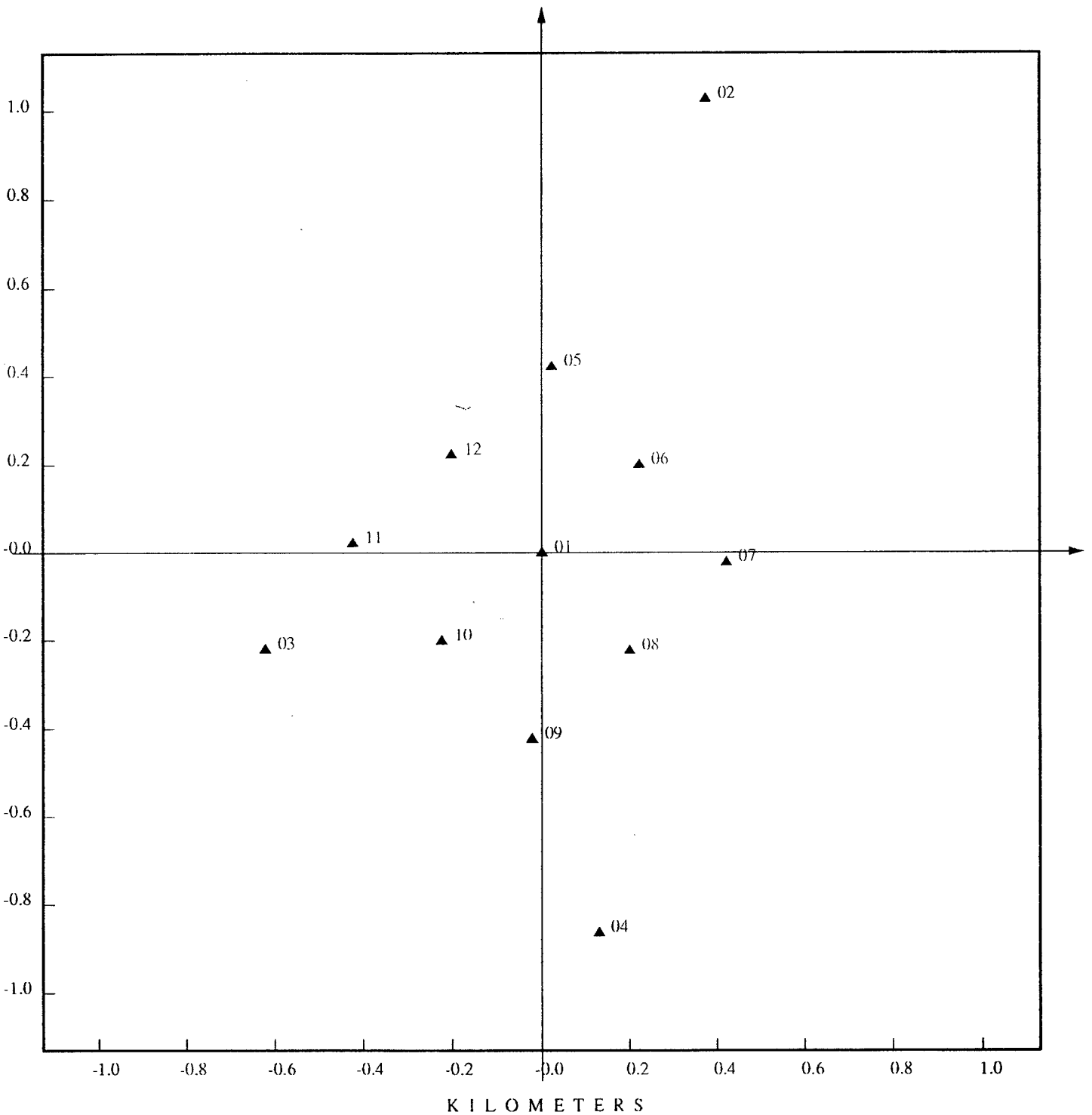
1. The SNDA package provides the powerful tools for multi-variant processing a great volume of real and synthetic multichannel data. In parallel with a wide range of the graphic and data handling facilities the package includes a rich variety of specialized data analysis procedures.

2. The SNDA package at this stage of development possesses the full capacity for a routine seismic monitoring data processing. At the same time, the SNDA system (as it following from its main concept) is open to future progress and incorporating new applications.

3. The adaptive group filtering algorithms provide a high suppression rate of broad band noise components of array records and high capacity for detecting and parameter estimating weak signals which amplitudes are tens times less than a noise level.

4. Job Control Language realized in SNDA package is the simple and convenient tool to design the high technology investigation projects, to perform numerical modeling, to assess, select and perform the optimal variant of data analysis in the automatic processing mode.

ORGH01 NH__02 SWH_03 SEH_04 A22_05 B32_06
C22_07 D32_08 E22_09 F32_10 G22_11 H32_12



Configuration of seismic array - GEYOKCHA_WBSA

Fig. 10.1

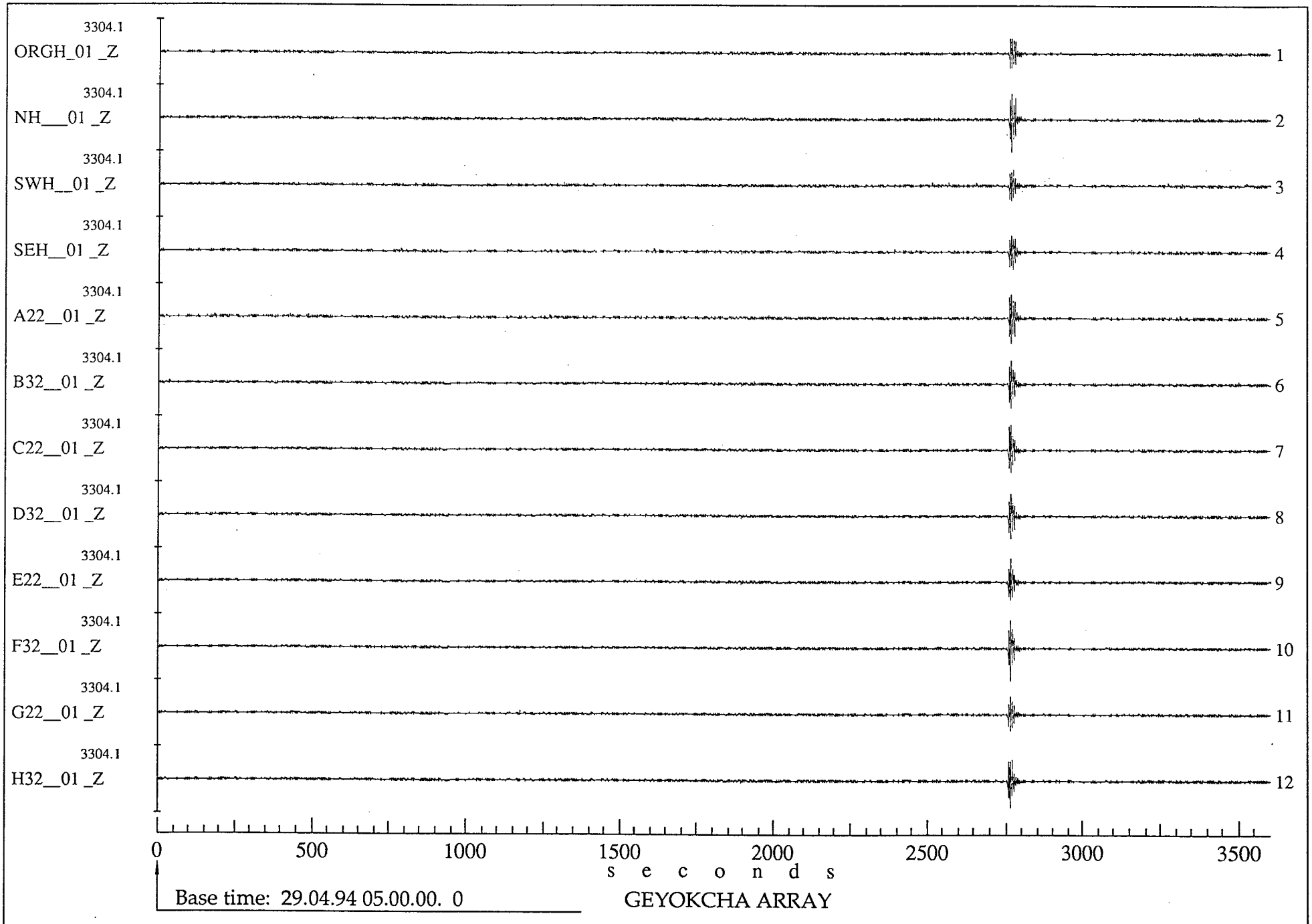

3304.1
ORGH_01 _Z
1
3304.1
NH___01 _Z
2
3304.1
SWH__01 _Z
3
3304.1
SEH__01 _Z
4
3304.1
A22__01 _Z
5
3304.1
B32__01 _Z
6
3304.1
C22__01 _Z
7
3304.1
D32__01 _Z
8
3304.1
E22__01 _Z
9
3304.1
F32__01 _Z
10
3304.1
G22__01 _Z
11
3304.1
H32__01 _Z
12
0
500
1000
1500
2000
2500
3000
3500
s e c o n d s
Base time: 29.04.94 05.00.00. 0
GEYOKCHA ARRAY
Wed May 24 14:22:30 1995 SYNAPSE Science Center

Fig. 10.2a

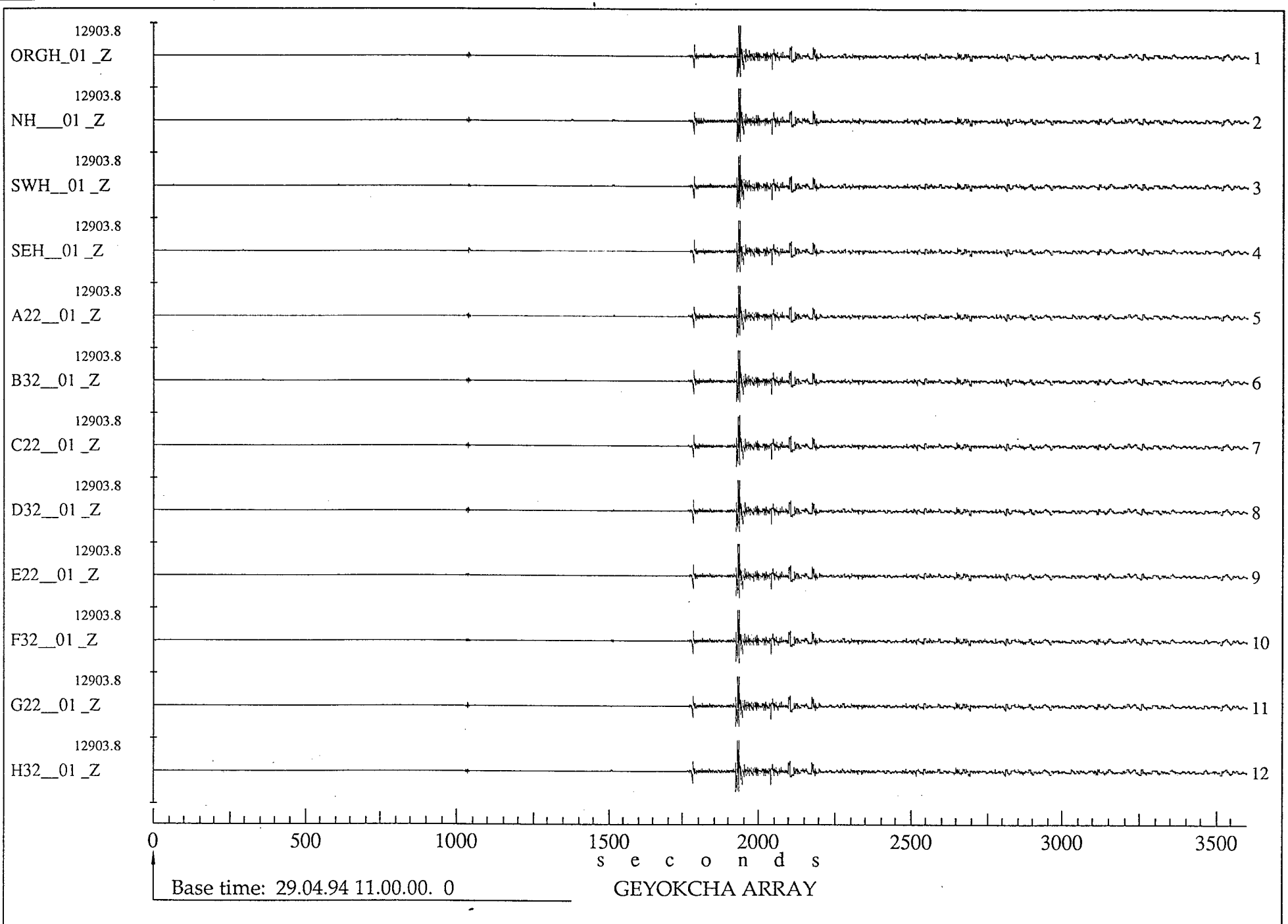

12903.8
ORGH_01 _Z
1
12903.8
NH___01 _Z
2
12903.8
SWH__01 _Z
3
12903.8
SEH__01 _Z
4
12903.8
A22__01 _Z
5
12903.8
B32__01 _Z
6
12903.8
C22__01 _Z
7
12903.8
D32__01 _Z
8
12903.8
E22__01 _Z
9
12903.8
F32__01 _Z
10
12903.8
G22__01 _Z
11
12903.8
H32__01 _Z
12
0
500
1000
1500
2000
2500
3000
3500
s e c o n d s
Base time: 29.04.94 11.00.00. 0
GEYOKCHA ARRAY
Wed May 24 14:24:55 1995 SYNAPSE Science Center

Fig. 10.2b

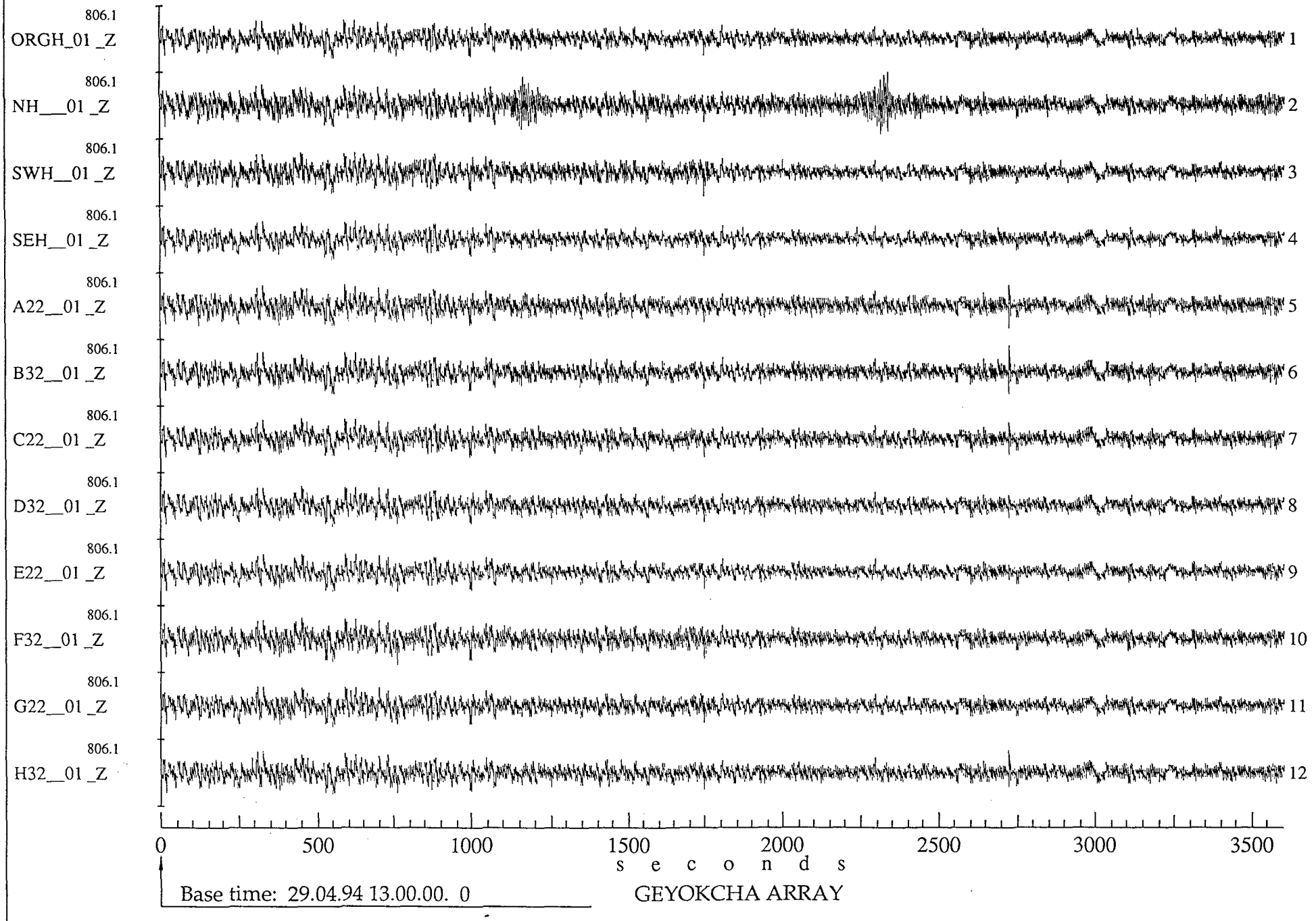

806.1
ORGH_01 _Z
1
806.1
NH___01 _Z
2
806.1
SWH__01 _Z
3
806.1
SEH__01 _Z
4
806.1
A22__01 _Z
5
806.1
B32__01 _Z
6
806.1
C22__01 _Z
7
806.1
D32__01 _Z
8
806.1
E22__01 _Z
9
806.1
F32__01 _Z
10
806.1
G22__01 _Z
11
806.1
H32__01 _Z
12
0
500
1000
1500
2000
2500
3000
3500
s e c o n d s
Base time: 29.04.94 13.00.00. 0
GEYOKCHA ARRAY
Wed May 24 14:26:42 1995 SYNAPSE Science Center

Fig. 10.2c

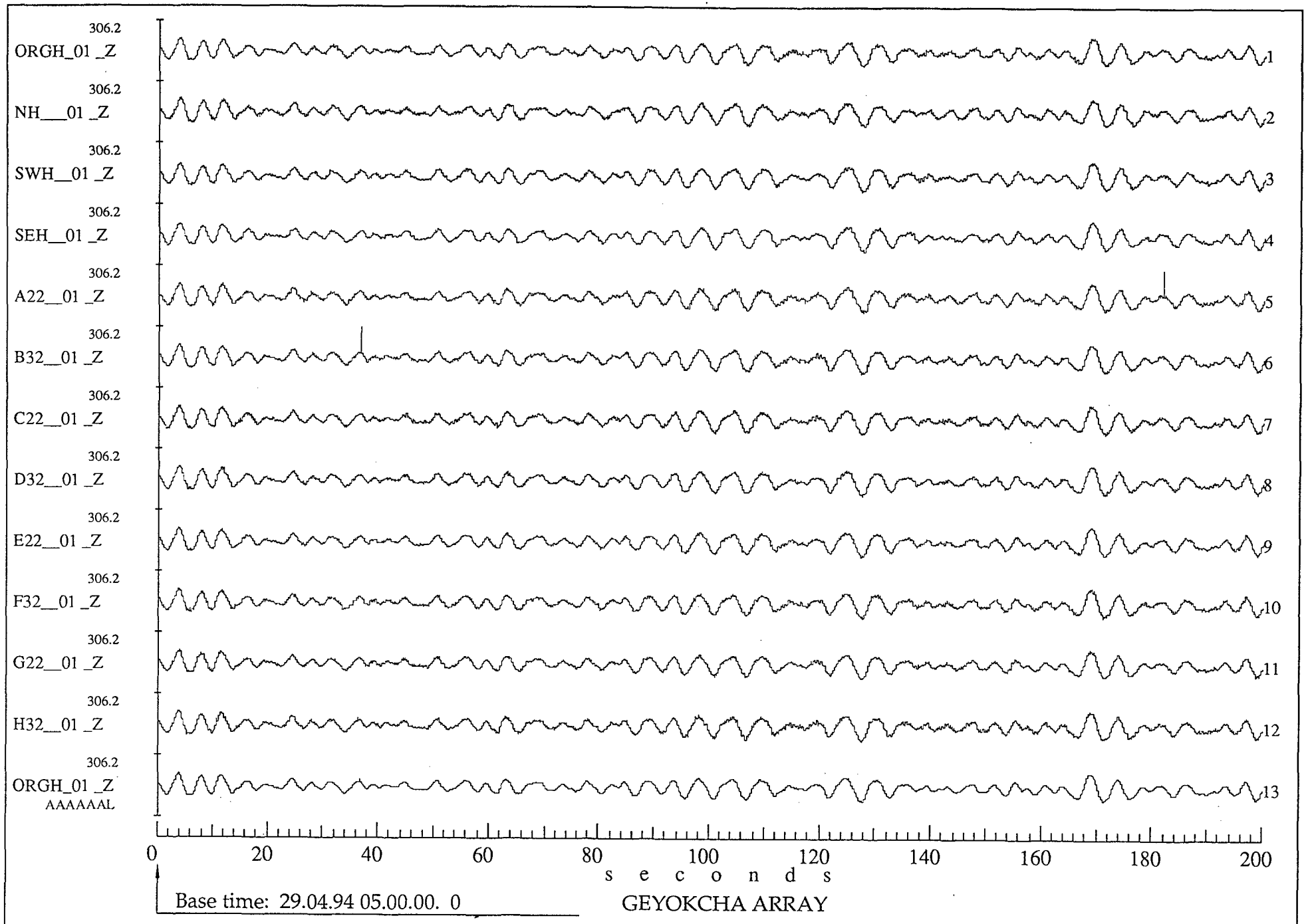

306.2
ORGH_01 _Z
306.2
NH___01 _Z
306.2
SWH__01 _Z
306.2
SEH__01 _Z
306.2
A22__01 _Z
306.2
B32__01 _Z
306.2
C22__01 _Z
306.2
D32__01 _Z
306.2
E22__01 _Z
306.2
F32__01 _Z
306.2
G22__01 _Z
306.2
H32__01 _Z
306.2
ORGH_01 _Z
AAAAAAL
1
2
3
4
5
6
7
8
9
10
11
12
13
0
20
40
60
80
100
120
140
160
180
200
s e c o n d s
Base time: 29.04.94 05.00.00. 0
GEYOKCHA ARRAY
Tue May 23 22:58:14 1995 SYNAPSE Science Center

Fig. 10.3a

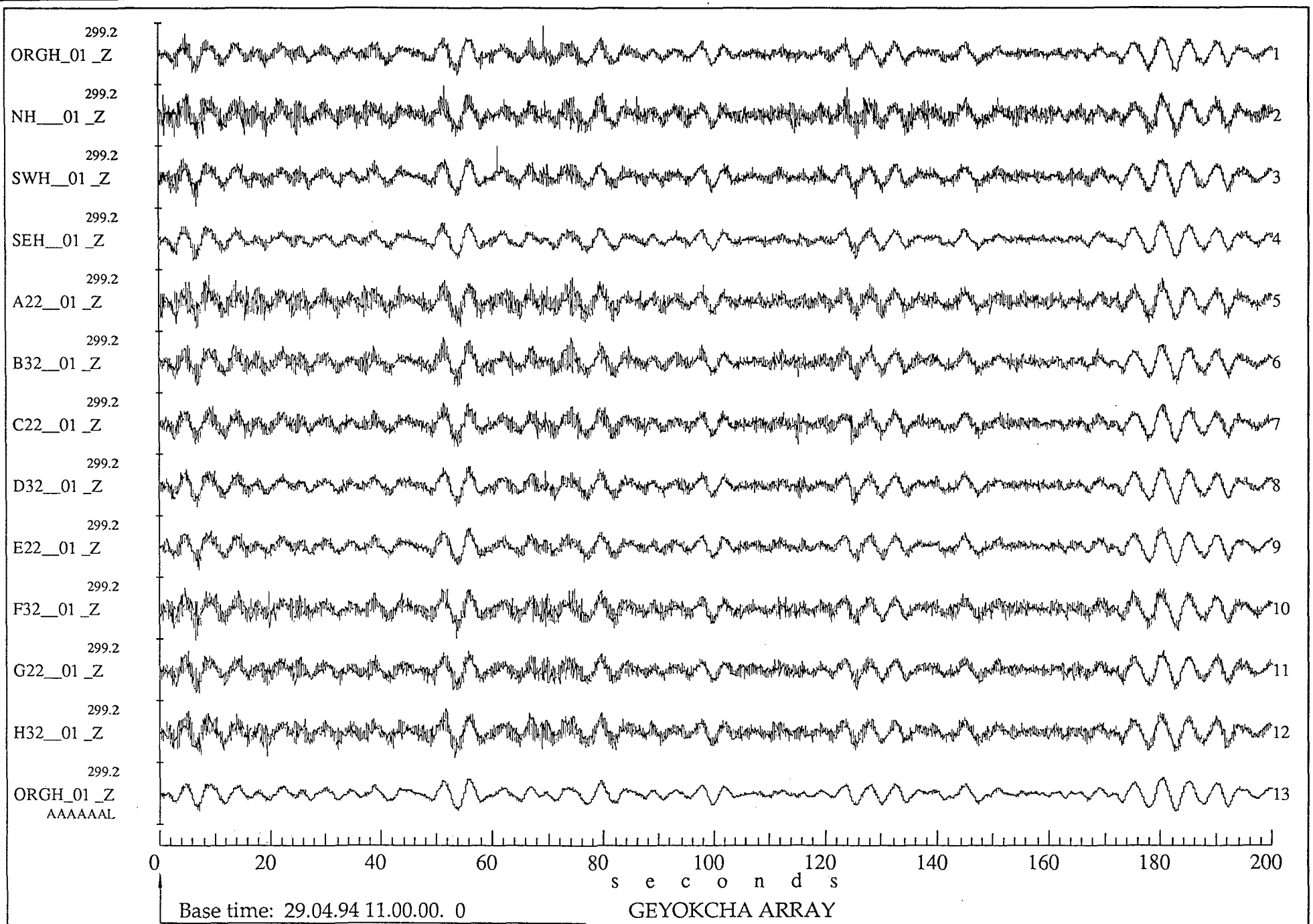
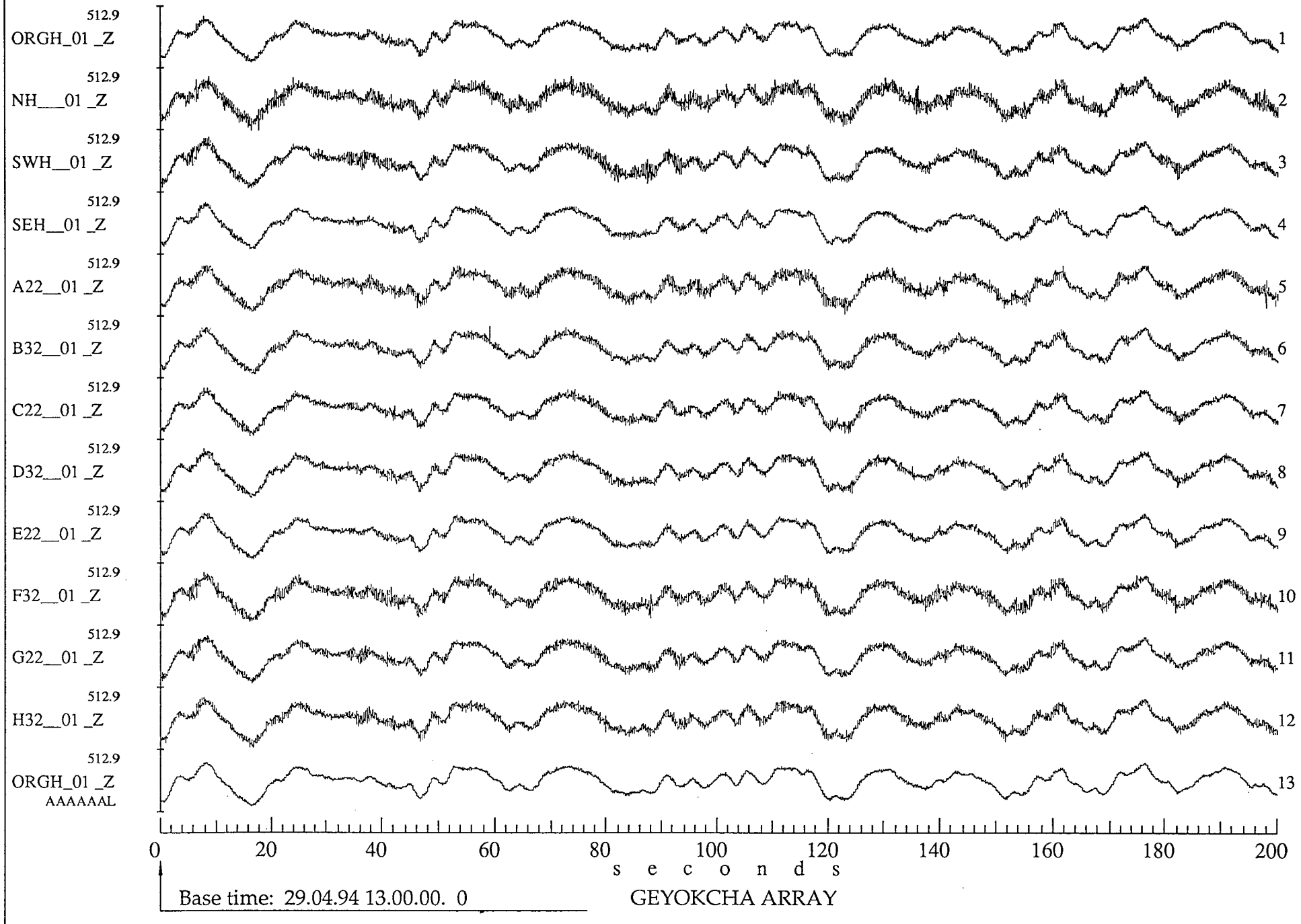

299.2
ORGH_01 _Z
1
299.2
NH___01 _Z
2
299.2
SWH__01 _Z
3
299.2
SEH__01 _Z
4
299.2
A22__01 _Z
5
299.2
B32__01 _Z
6
299.2
C22__01 _Z
7
299.2
D32__01 _Z
8
299.2
E22__01 _Z
9
299.2
F32__01 _Z
10
299.2
G22__01 _Z
11
299.2
H32__01 _Z
12
299.2
ORGH_01 _Z
AAAAAAL
13
0
20
40
60
80
100
120
140
160
180
200
s e c o n d s
Base time: 29.04.94 11.00.00. 0
GEYOKCHA ARRAY
Tue May 23 23:07:03 1995 SYNAPSE Science Center

Fig. 10.3b



Tue May 23 23:11:54 1995 SYNAPSE Science Center

Fig. 10.3c

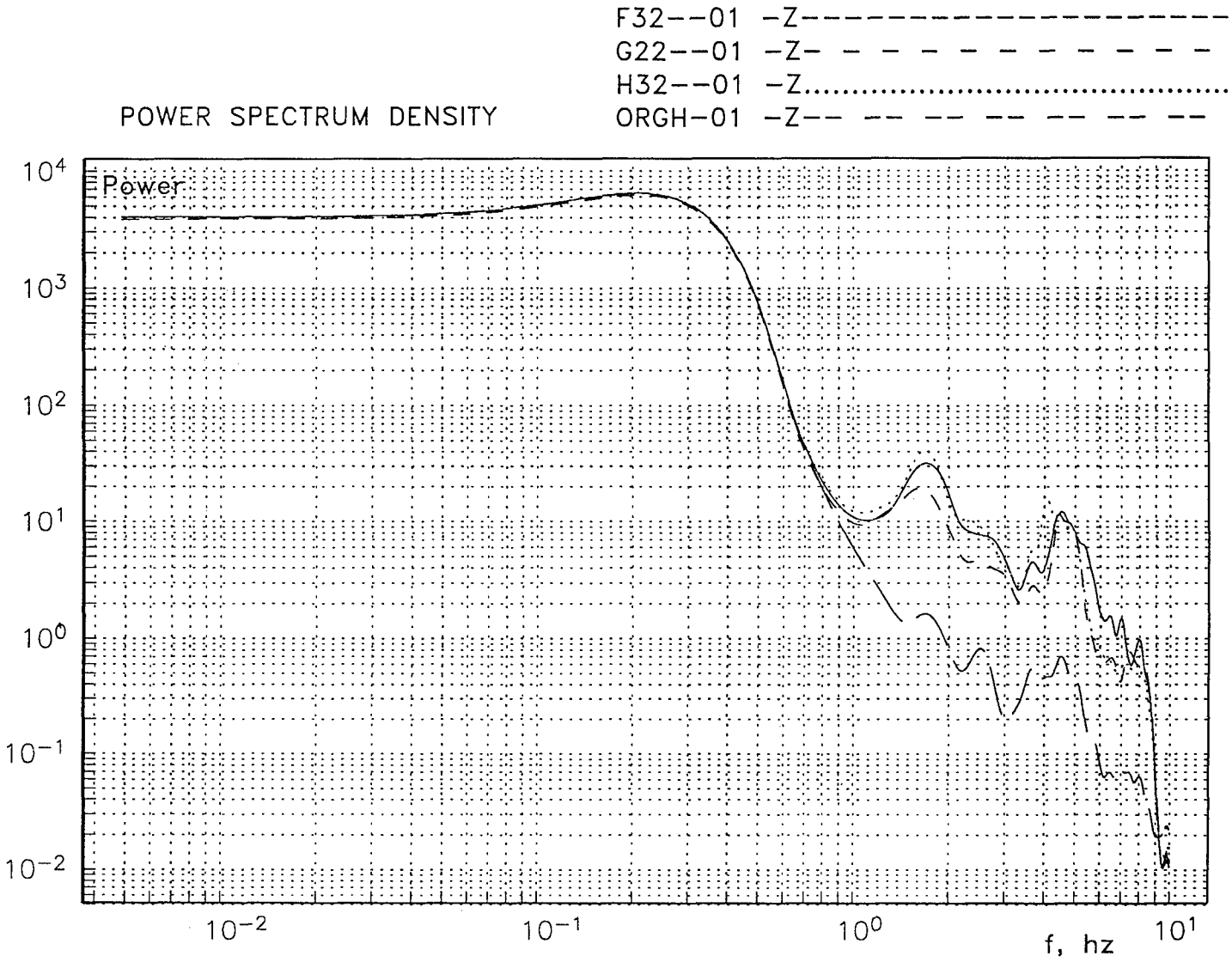

F32--01 -Z----------
G22--01 -Z- - - - - - - - - - -
H32--01 -Z.....
ORGH-01 -Z-- -- -- -- -- -- --
POWER SPECTRUM DENSITY
Power
10^4
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-2}
10^{-1}
10^0
10^1
f, hz

Fig. 10.4a

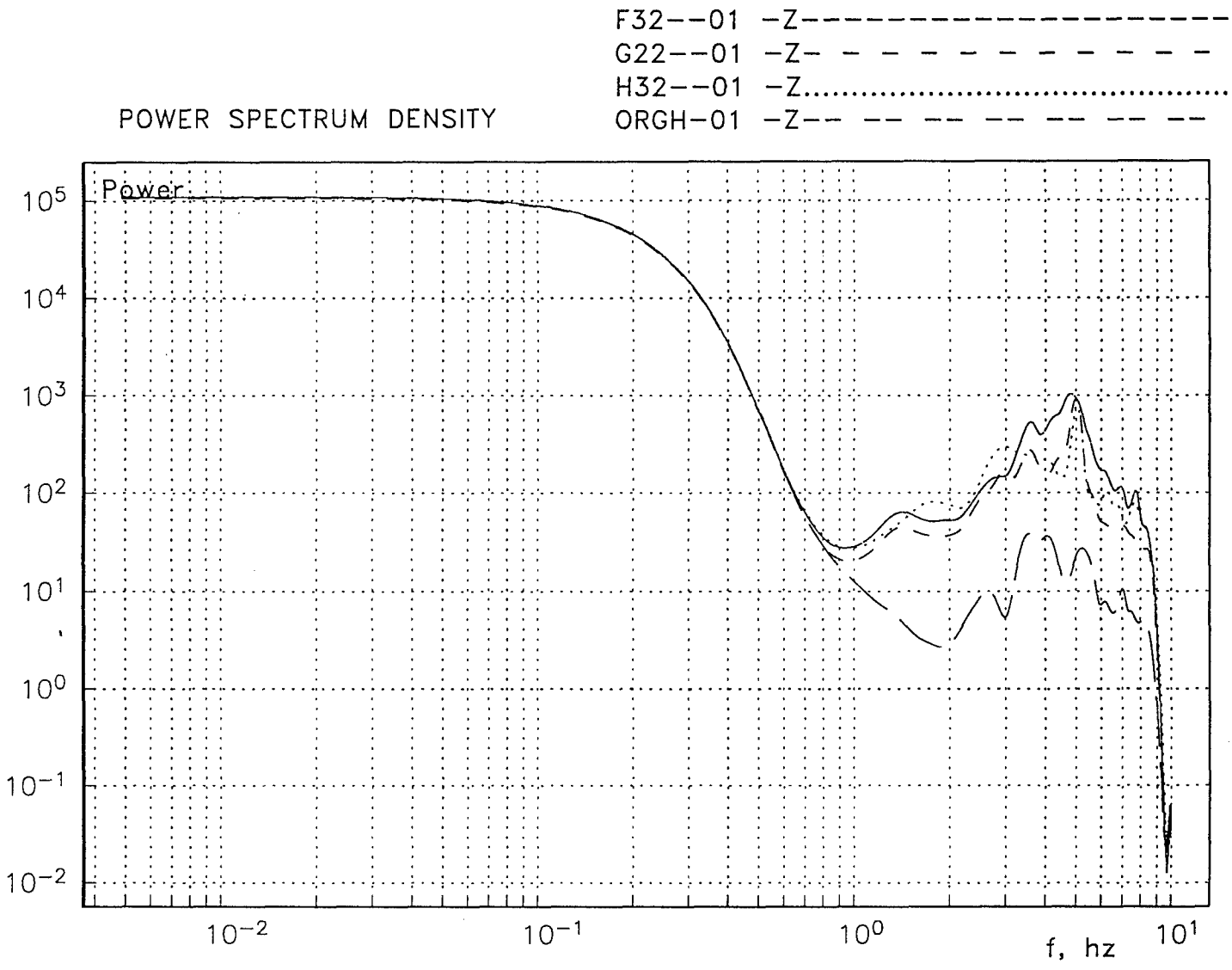

F32--01 -Z----------
G22--01 -Z- - - - - - - - - -
H32--01 -Z.....
ORGH-01 -Z-- -- -- -- -- -- --
POWER SPECTRUM DENSITY
Power
10^5
10^4
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-2}
10^{-1}
10^0
10^1
f, hz

Fig. 10.4b

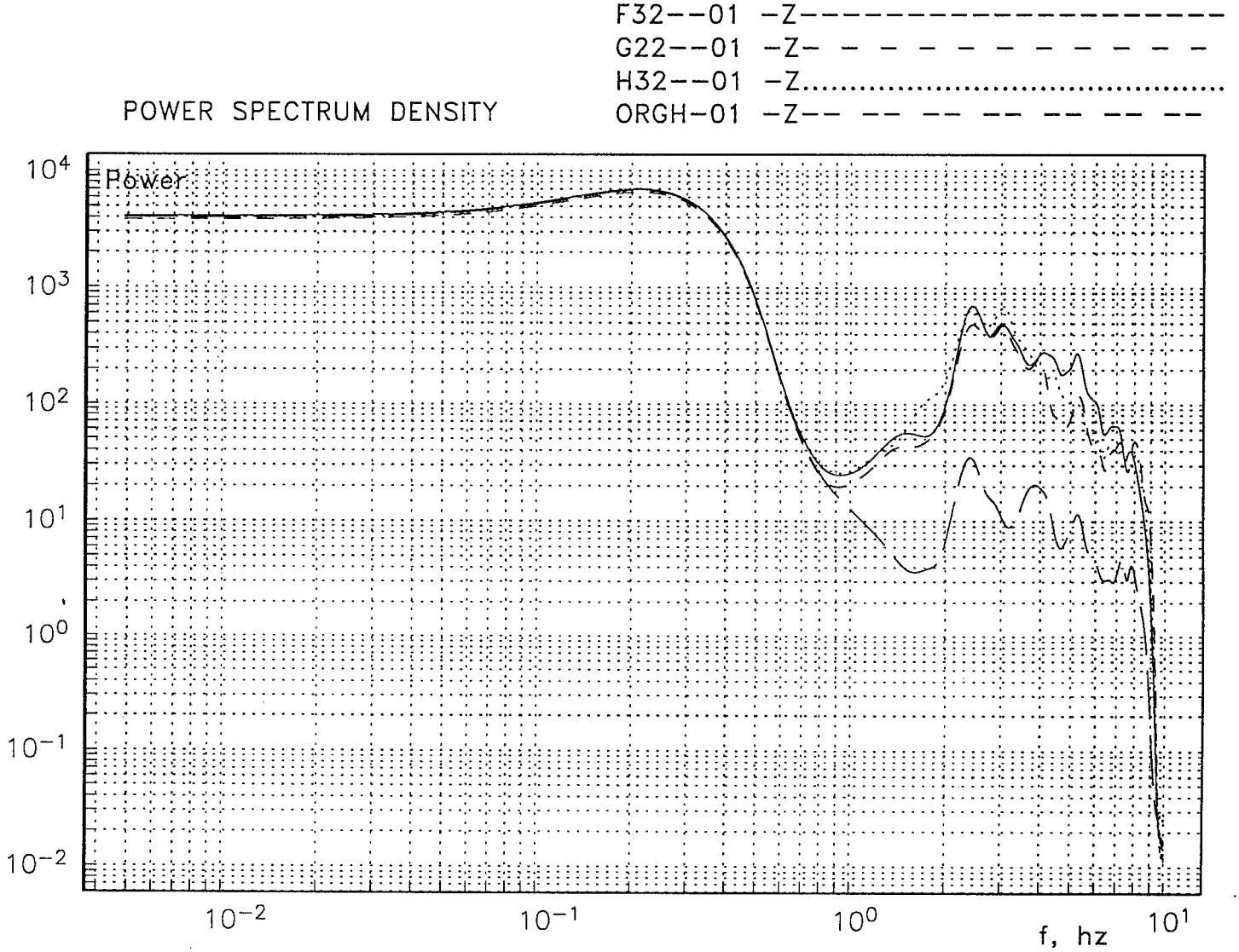
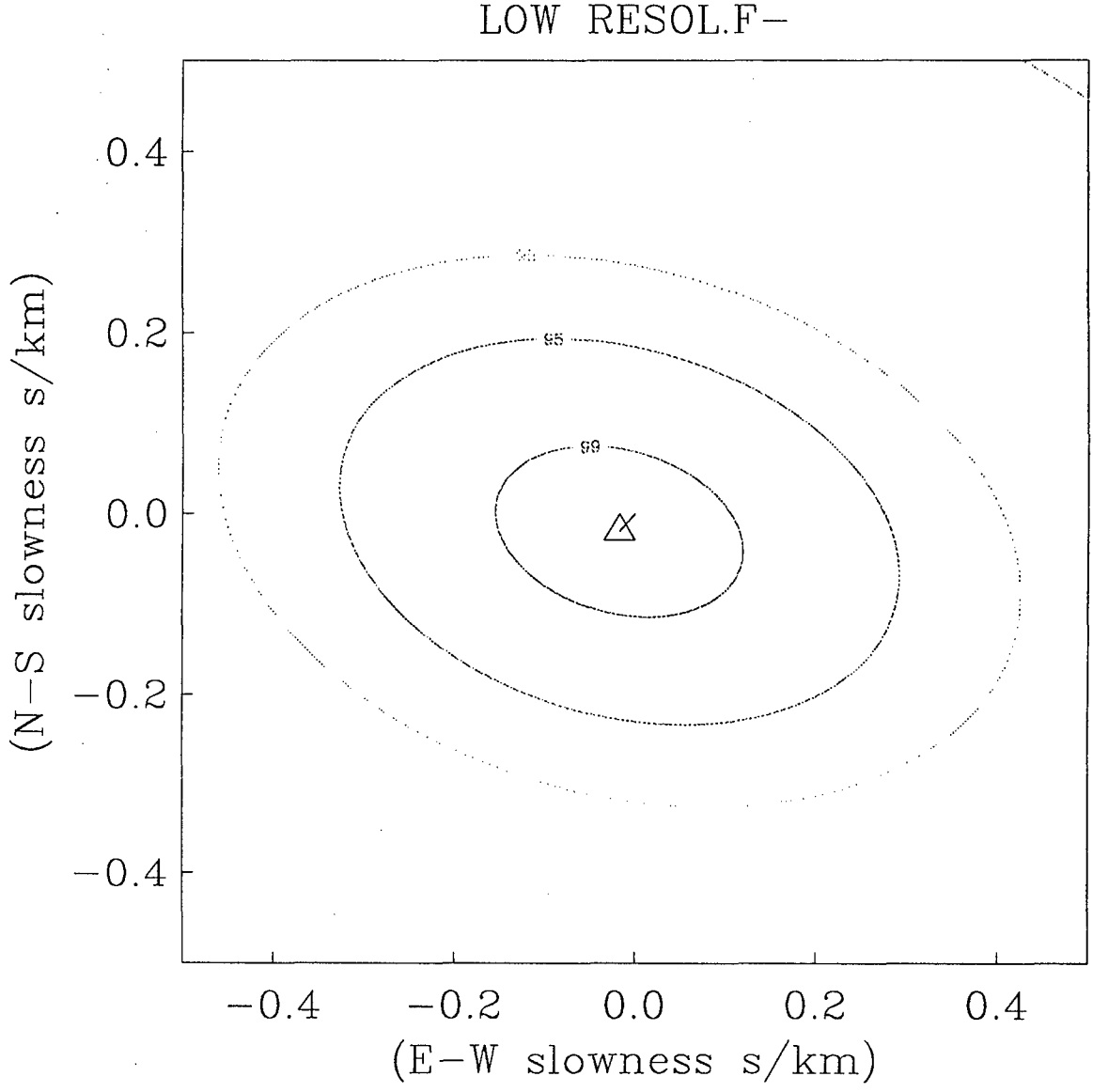

F32--01 -Z----------
G22--01 -Z- - - - - - - - - - -
H32--01 -Z.....
ORGH-01 -Z-- -- -- -- -- -- --
POWER SPECTRUM DENSITY
Power
10^4
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-2}
10^{-1}
10^0
10^1
f, hz

Fig. 10.4c



Frequency = 0.4 Hz, LOW RESILUTION Azim. of max= 221.87 Ap. vel. of max= 38.43 Adapt. AR order IP=10 Adapt. MA order IQ=10 Regulariz.=.0001000 Adapt. mode =−4

Fig. 10.5

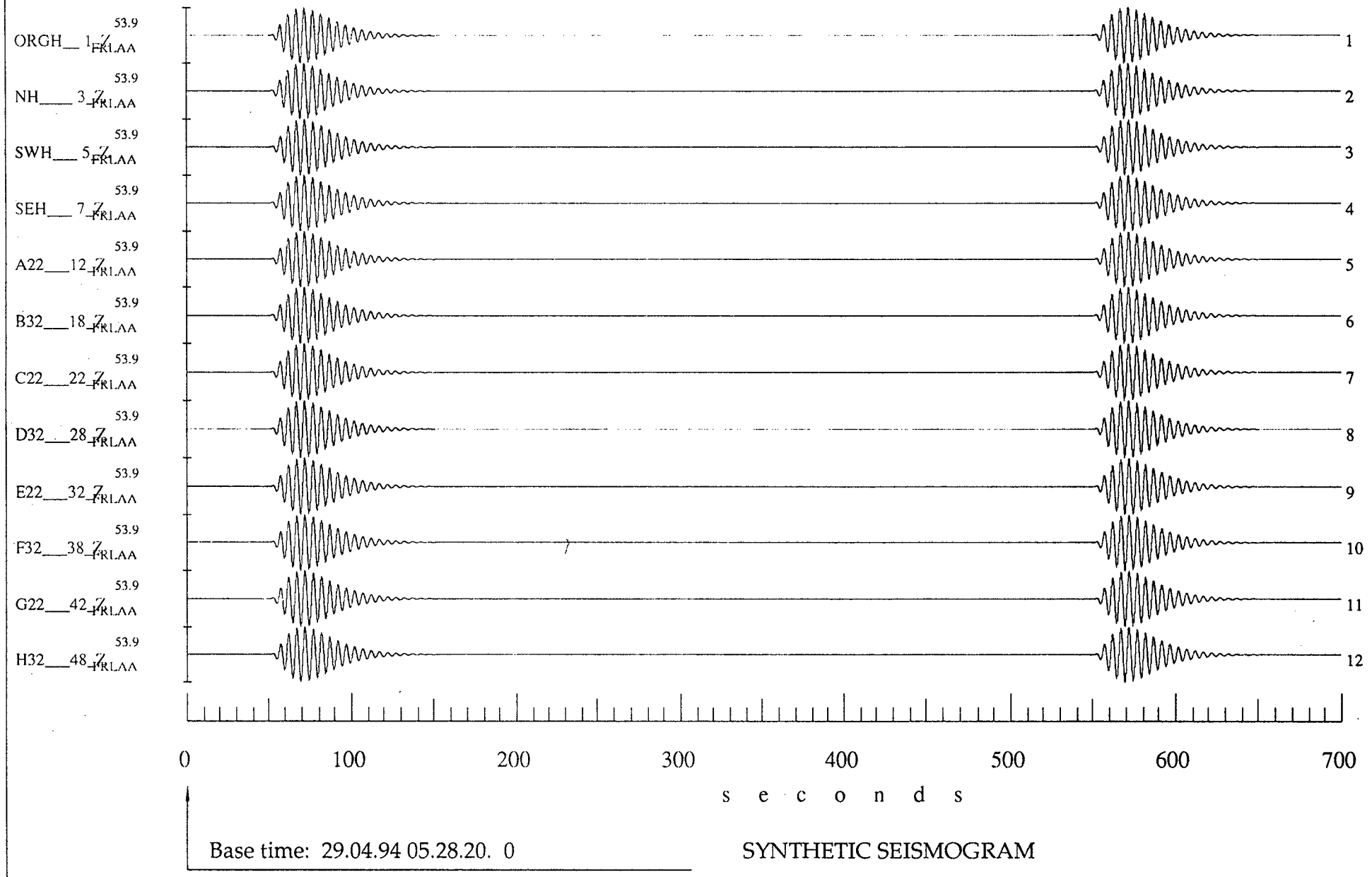
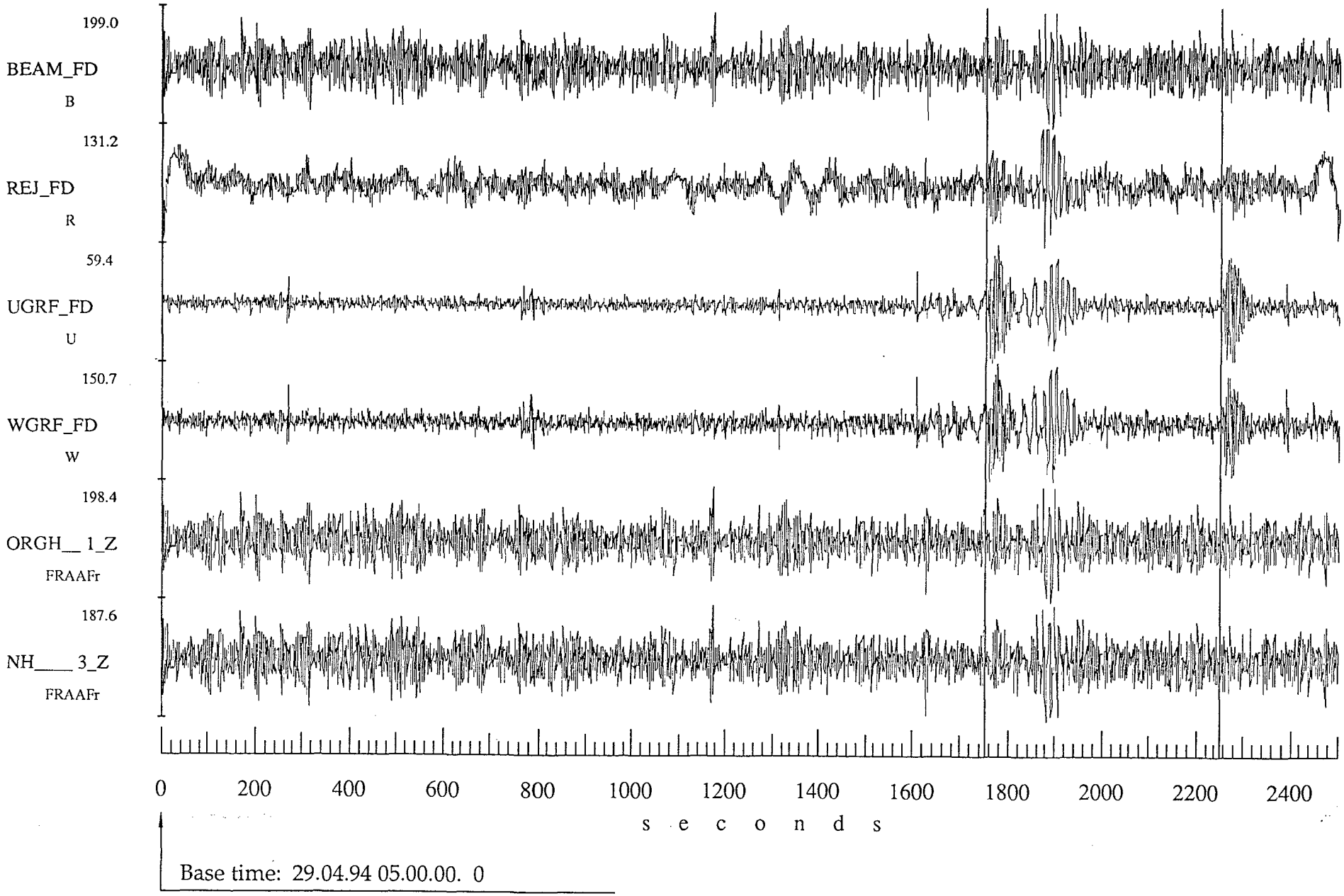
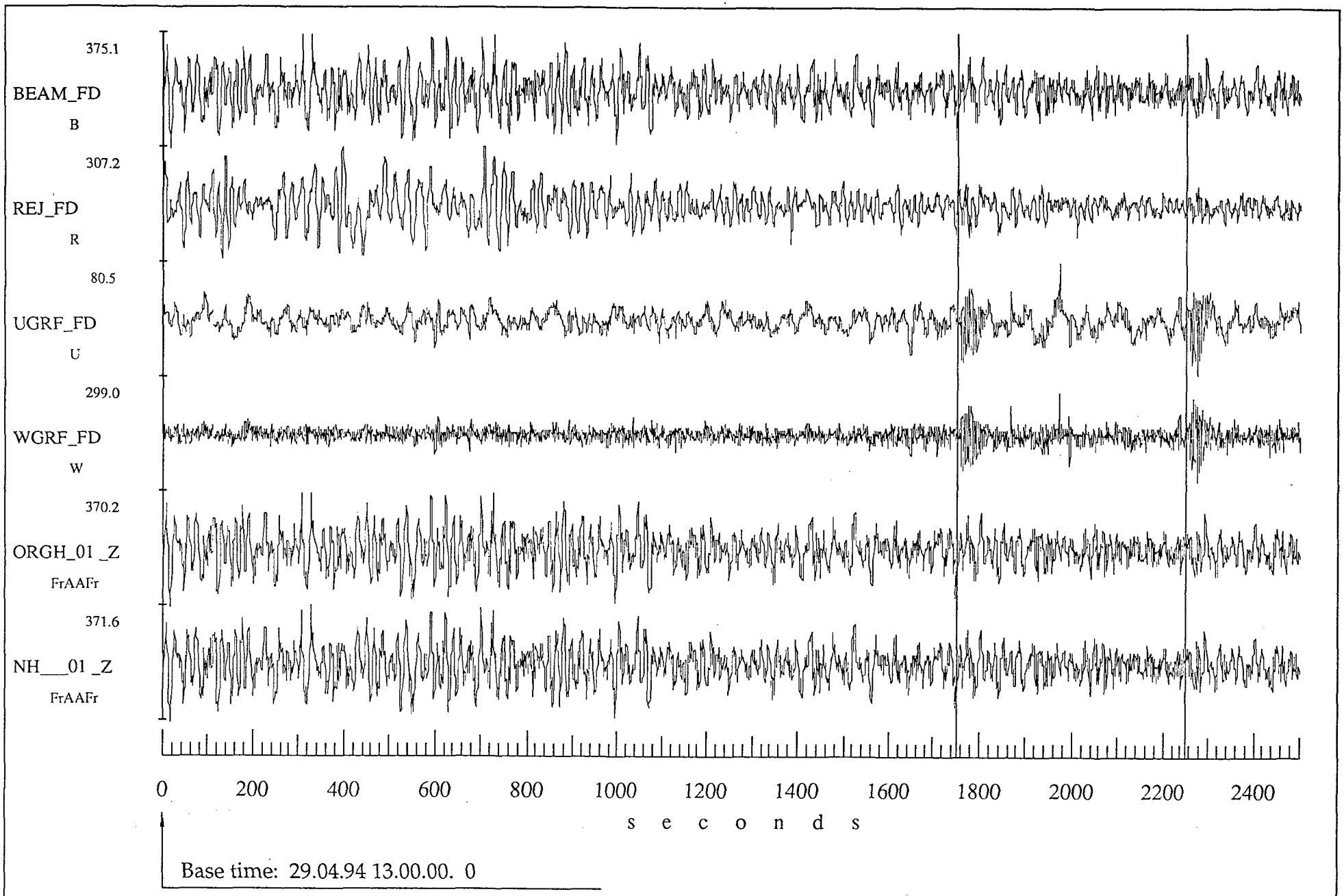

ORGH__ 1_Z 53.9 FRI.AA
NH____ 3_Z 53.9 FRI.AA
SWH___ 5_Z 53.9 FRI.AA
SEH___ 7_Z 53.9 FRI.AA
A22___12_Z 53.9 FRI.AA
B32___18_Z 53.9 FRI.AA
C22___22_Z 53.9 FRI.AA
D32___28_Z 53.9 FRI.AA
E22___32_Z 53.9 FRI.AA
F32___38_Z 53.9 FRI.AA
G22___42_Z 53.9 FRI.AA
H32___48_Z 53.9 FRI.AA
1
2
3
4
5
6
7
8
9
10
11
12
0
100
200
300
400
500
600
700
s e c o n d s
Base time: 29.04.94 05.28.20. 0
SYNTHETIC SEISMOGRAM

Fig. 10.6



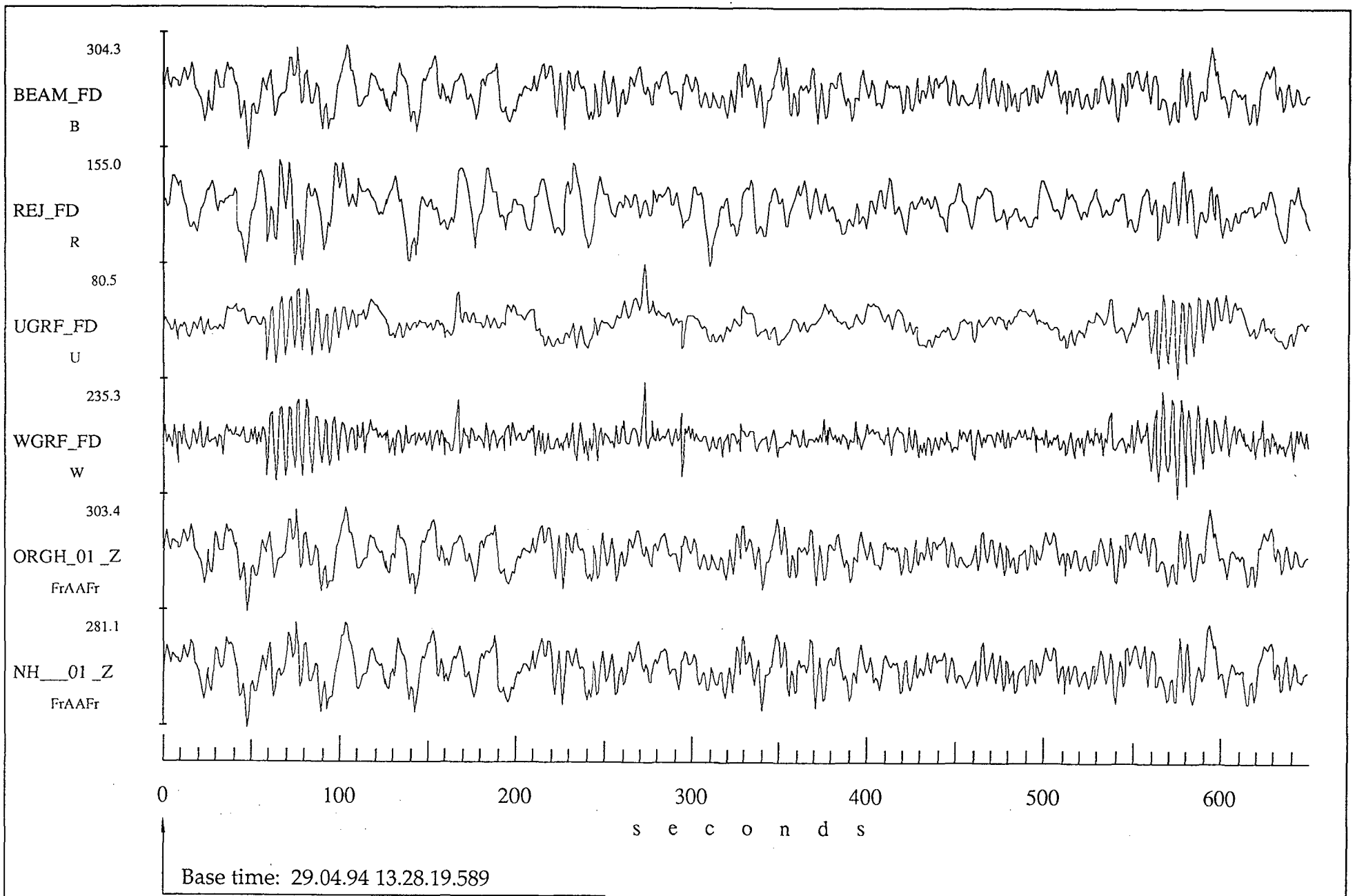
Tue May 23 20:38:34 1995 SYNAPSE Science Center

Fig. 10.7a



Tue May 23 21:19:13 1995 SYNAPSE Science Center

Fig. 10.7b



Tue May 23 21:21:26 1995 SYNAPSE Science Center

Fig. 10.7c

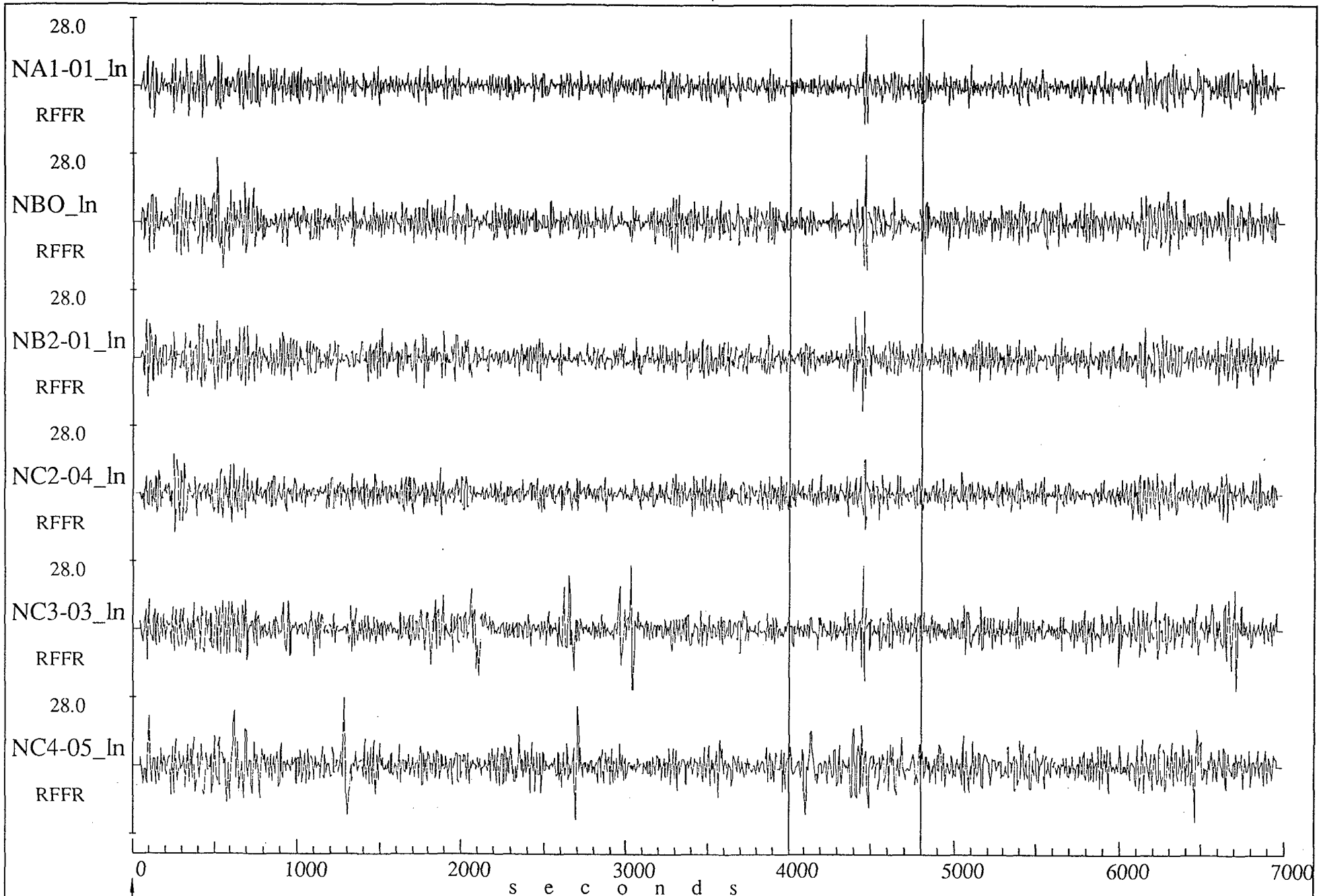

28.0
NA1-01_ln
RFFR
28.0
NBO_ln
RFFR
28.0
NB2-01_ln
RFFR
28.0
NC2-04_ln
RFFR
28.0
NC3-03_ln
RFFR
28.0
NC4-05_ln
RFFR
0
1000
2000
3000
4000
5000
6000
7000
s e c o n d s
Base time: 30.05.83 03.40.00. 0
NORSAR, EV1,n, NUCLEAR EXPLOSION SURFACE WAVE
WHOLE DATA INTERVAL, FB< 0.1 HZ
Mon May 22 17:41:55 1995 SYNAPSE Science Center

Fig. 10.8a

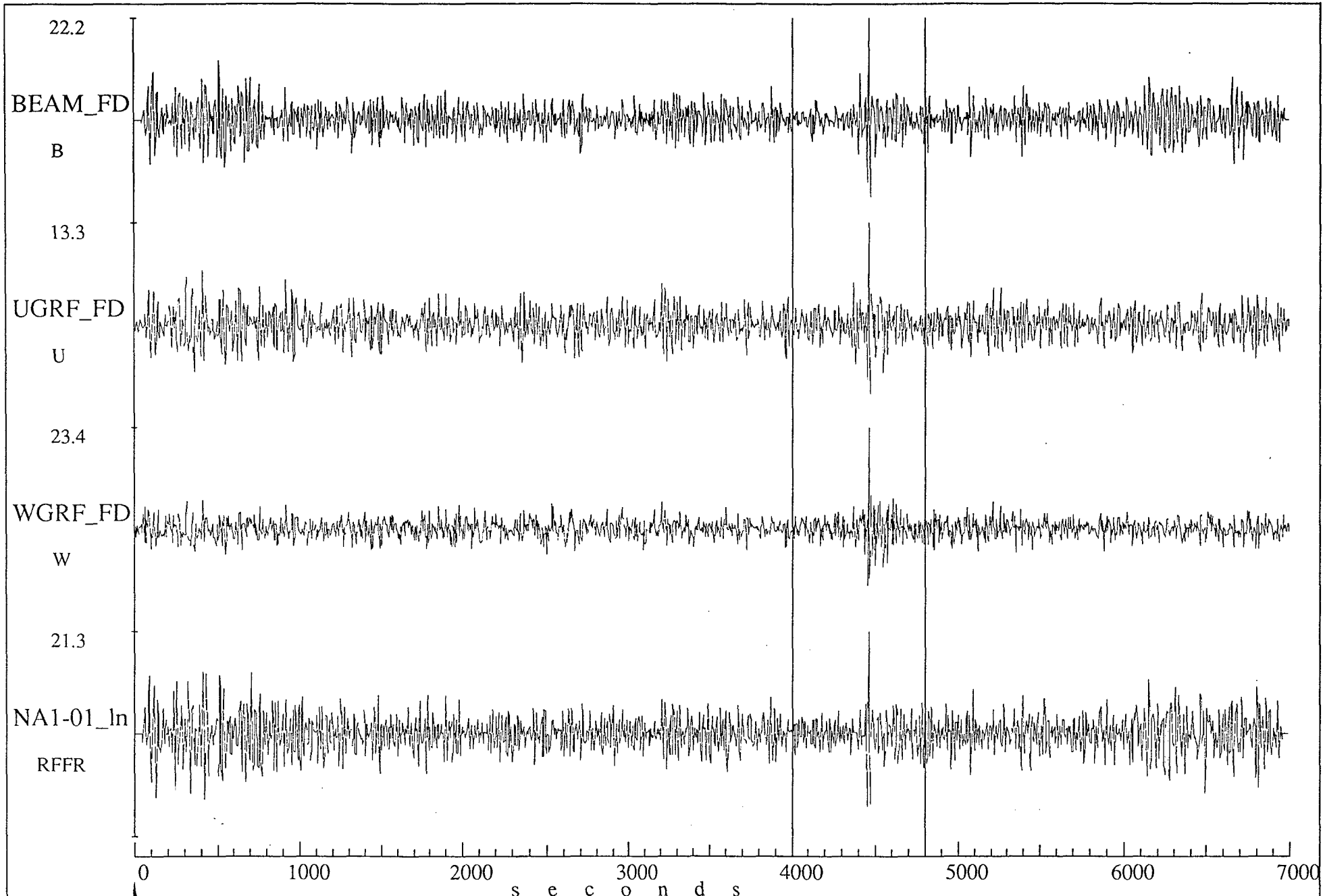

22.2
BEAM_FD
B
13.3
UGRF_FD
U
23.4
WGRF_FD
W
21.3
NA1-01_ln
RFFR
0
1000
2000
3000
4000
5000
6000
7000
s e c o n d s
Base time: 30.05.83 03.40.00. 0
NORSAR, EV1, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, MA-15
Mon May 22 17:42:44 1995 SYNAPSE Science Center

Fig. 10.8b

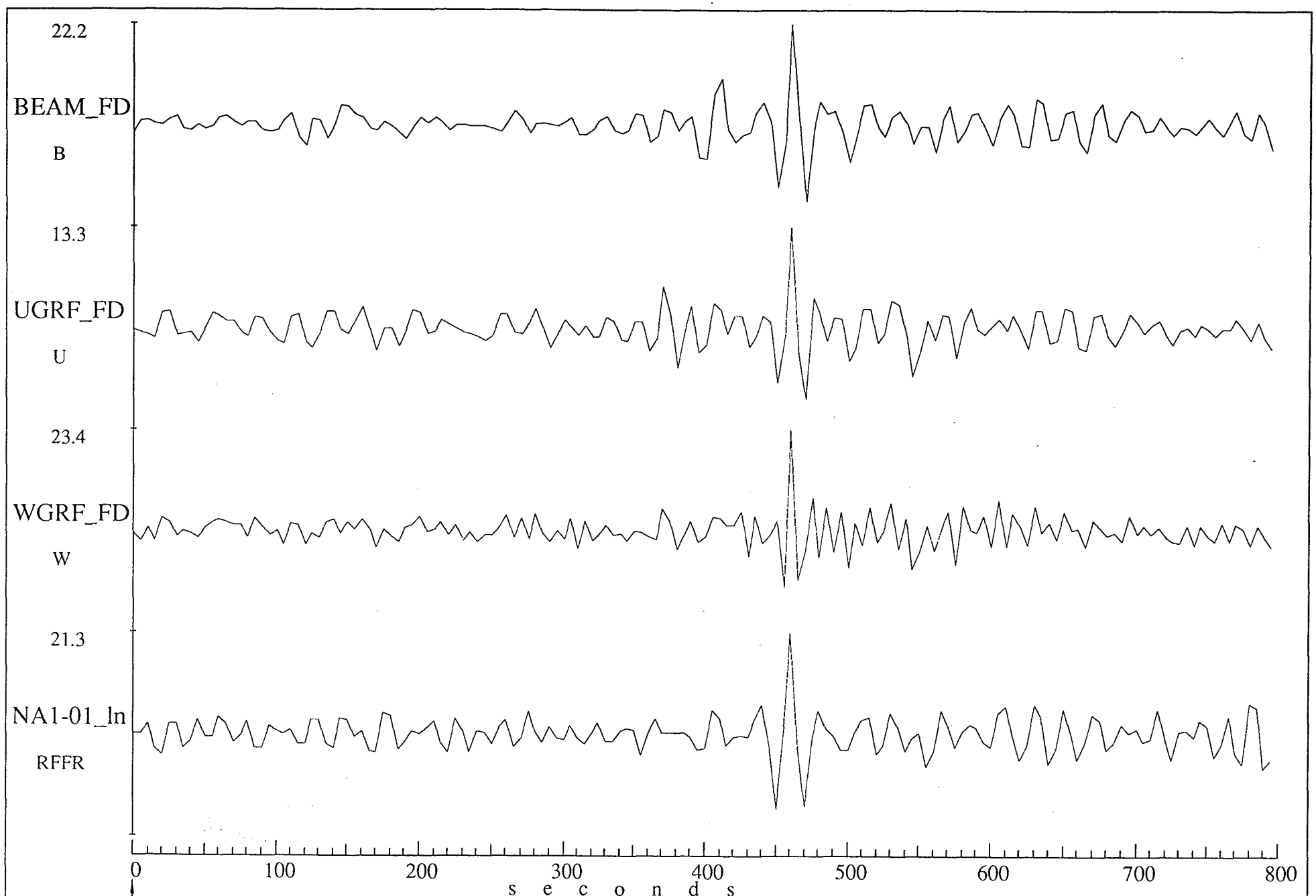

22.2
BEAM_FD
B
13.3
UGRF_FD
U
23.4
WGRF_FD
W
21.3
NA1-01_ln
RFFR
0
100
200
300
400
500
600
700
800
s e c o n d s
Base time: 30.05.83 04.46.40. 0
NORSAR, EV1, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, MA-15
Mon May 22 17:43:04 1995 SYNAPSE Science Center

Fig. 10.8 c

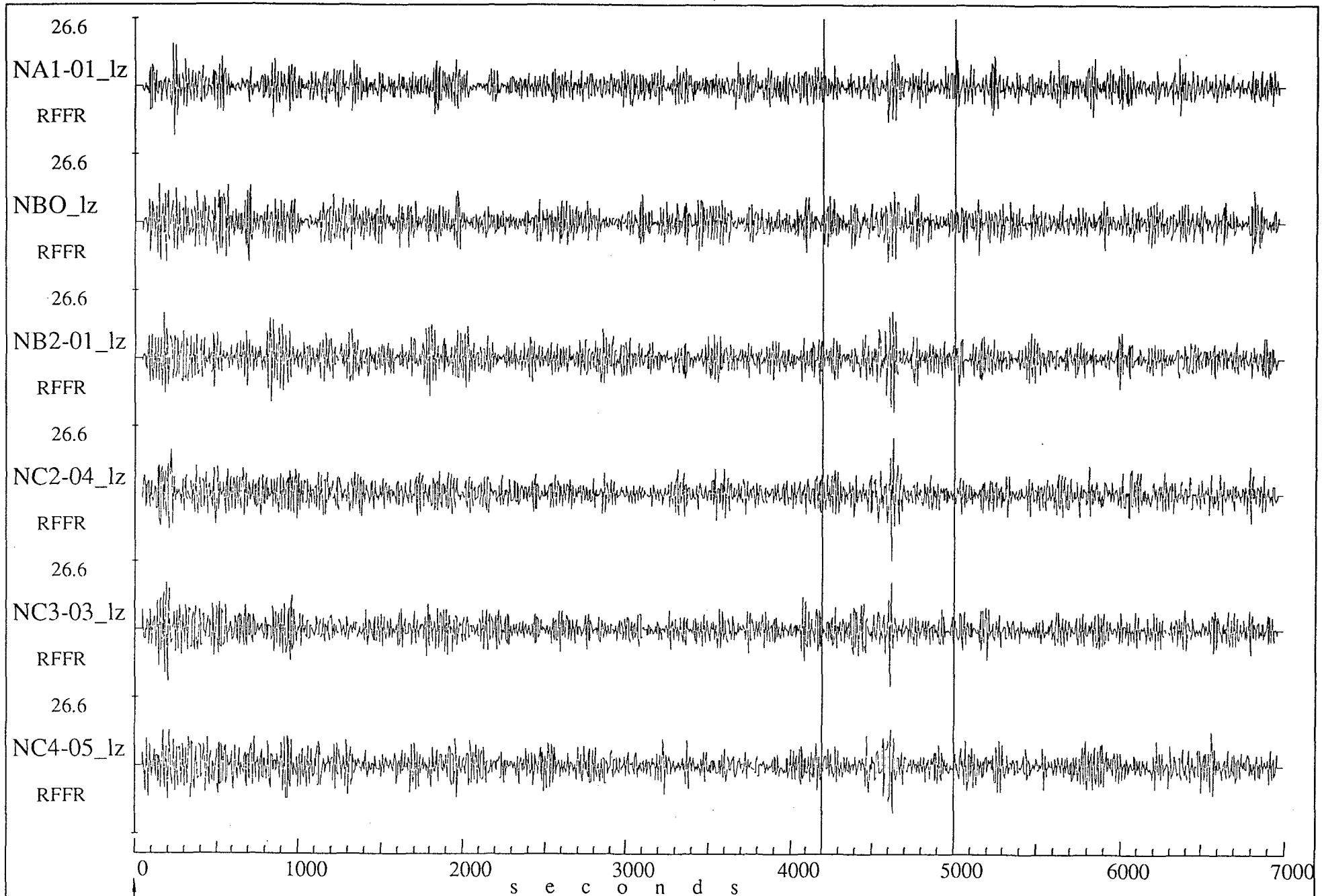

26.6
NA1-01_lz
RFFR
26.6
NBO_lz
RFFR
26.6
NB2-01_lz
RFFR
26.6
NC2-04_lz
RFFR
26.6
NC3-03_lz
RFFR
26.6
NC4-05_lz
RFFR
0
1000
2000
3000
4000
5000
6000
7000
s e c o n d s
Base time: 30.05.83 03.40.00. 0
NORSAR, EV1z, NUCLEAR EXPLOSION SURFACE WAVES
WHOLE DATA INTERVAL, FB< 0.1 HZ
Mon May 22 17:44:47 1995 SYNAPSE Science Center

Fig. 10.9a

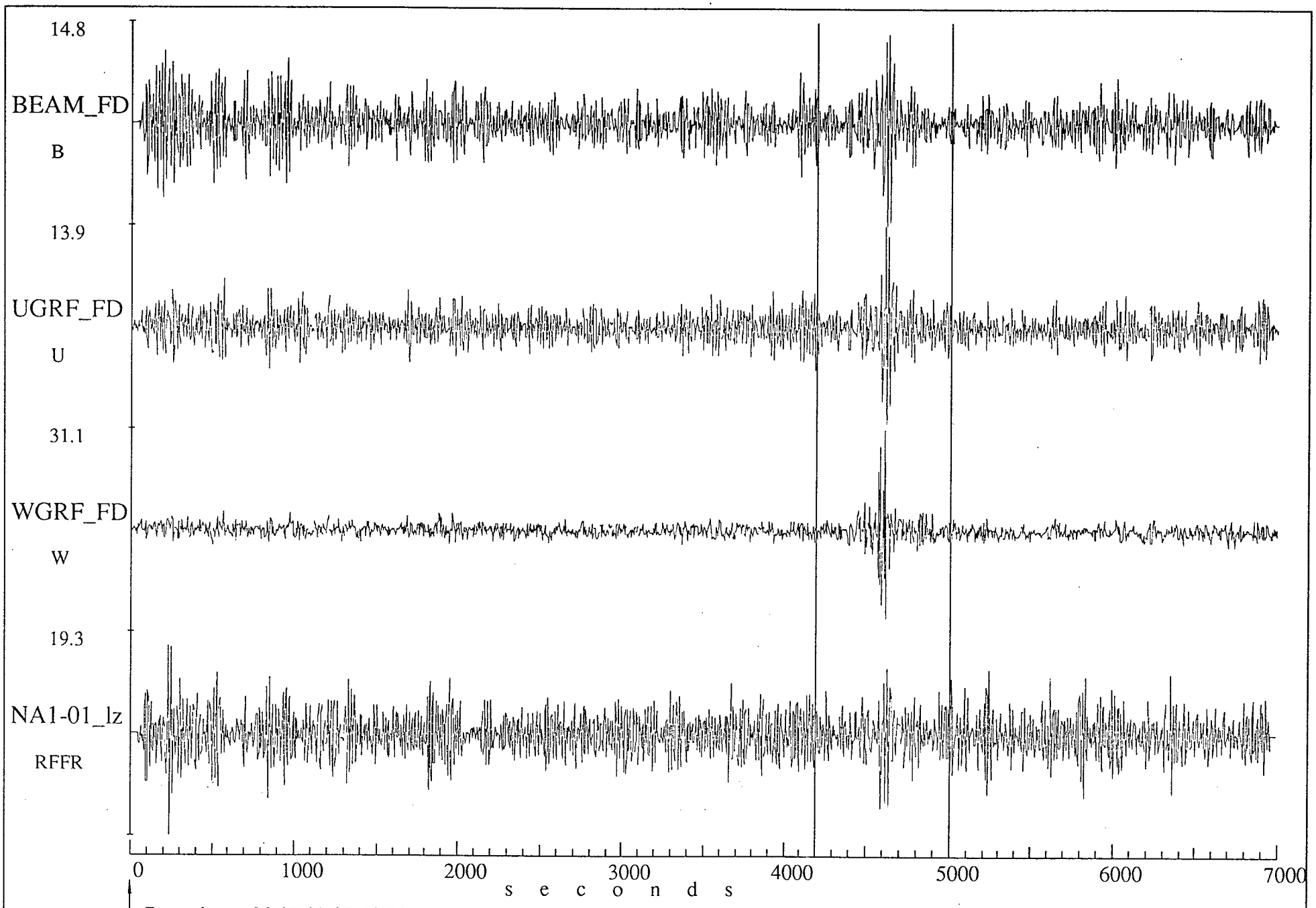
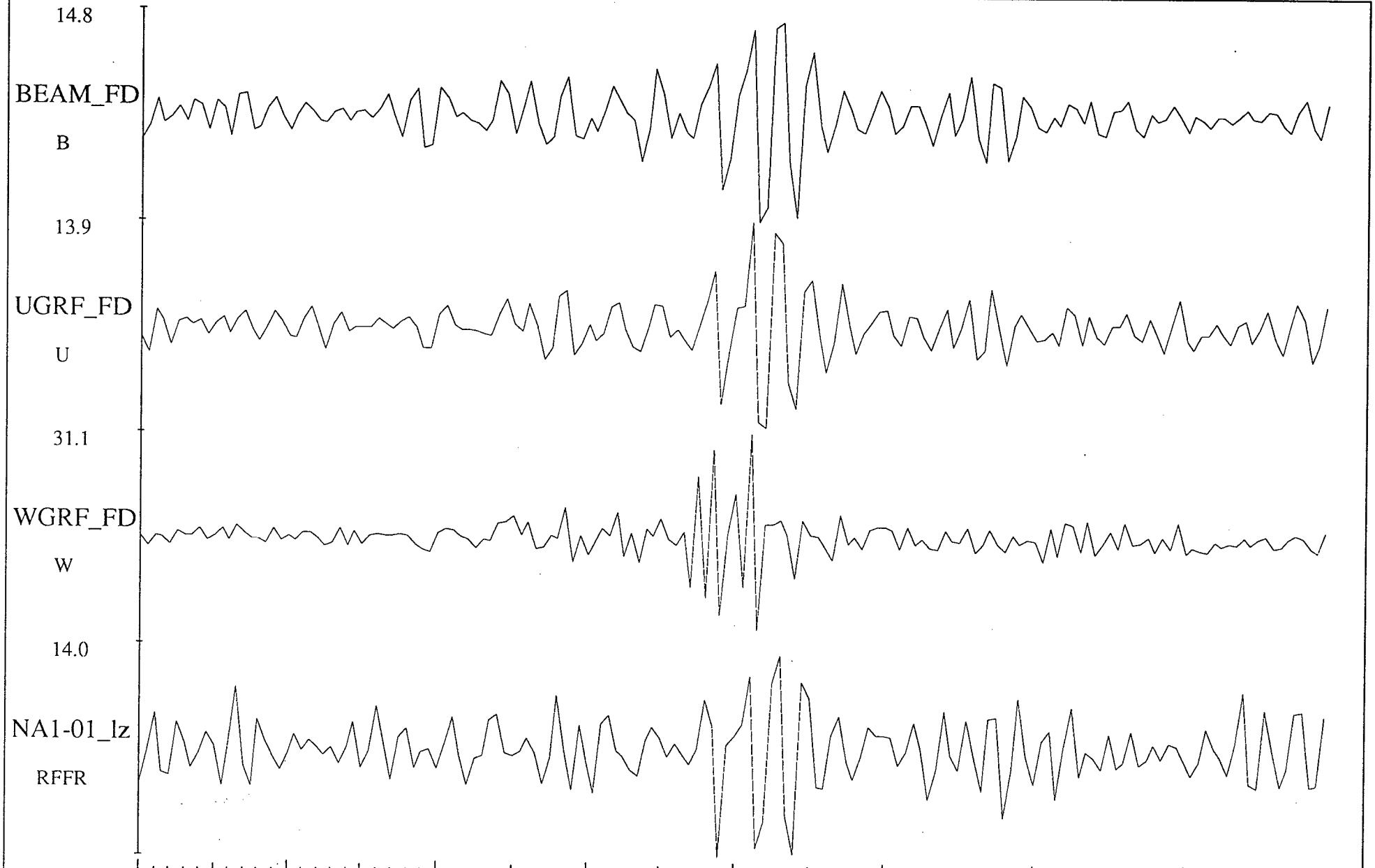

14.8
BEAM_FD
B
13.9
UGRF_FD
U
31.1
WGRF_FD
W
19.3
NA1-01_lz
RFFR
0
1000
2000
3000
4000
5000
6000
7000
s e c o n d s
Base time: 30.05.83 03.40.00. 0
NORSAR, EV1z, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, MA-15
Mon May 22 17:45:46 1995 SYNAPSE Science Center

Fig. 10.9b



NORSAR, EV1z, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, MA-15

Mon May 22 17:46:20 1995 SYNAPSE Science Center

Fig. 10.9c

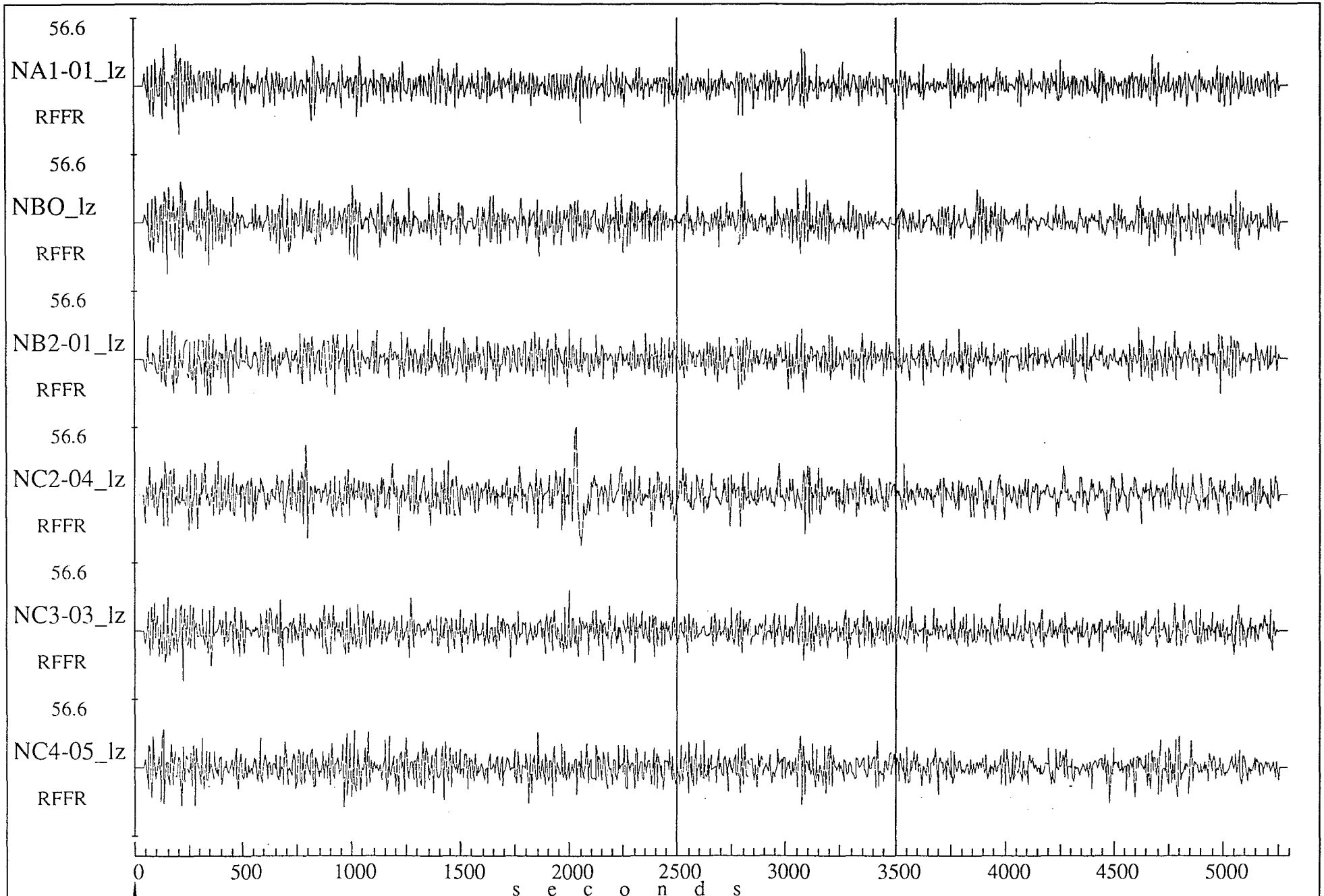

56.6
NA1-01_lz
RFFR
56.6
NBO_lz
RFFR
56.6
NB2-01_lz
RFFR
56.6
NC2-04_lz
RFFR
56.6
NC3-03_lz
RFFR
56.6
NC4-05_lz
RFFR
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
s e c o n d s
Base time: 27.10.84 06.30.00. 0
NORSAR, EV2,z, NUCLEAR EXPLOSION SURFACE WAVE
WHOLE DATA INTERVAL, FB< 0.1 HZ
Mon May 22 17:48:02 1995 SYNAPSE Science Center

Fig. 10.10a

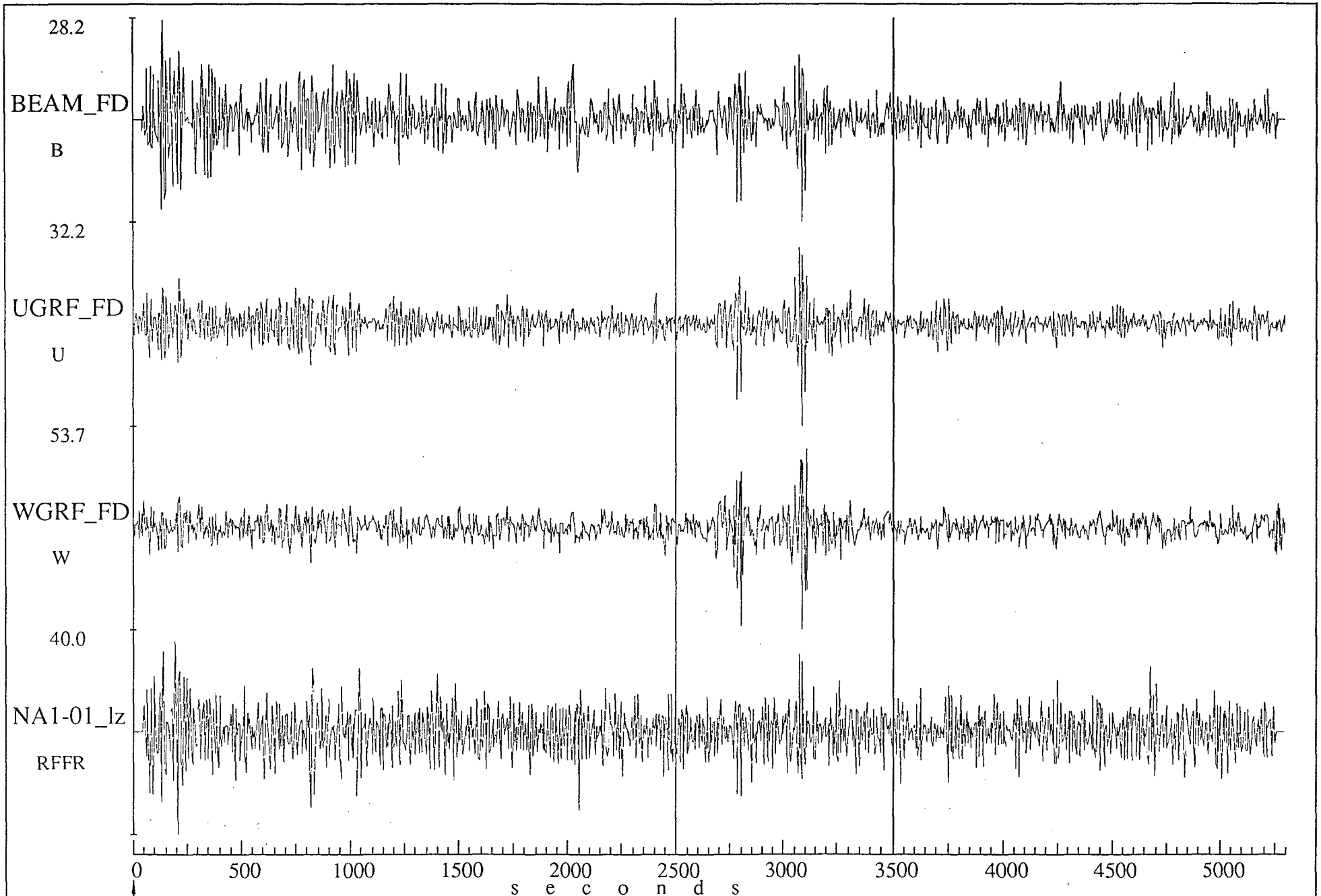

28.2
BEAM_FD
B
32.2
UGRF_FD
U
53.7
WGRF_FD
W
40.0
NA1-01_lz
RFFR
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
s e c o n d s
Base time: 27.10.84 06.30.00. 0
NORSAR, EV2,z, NUCLEAR EXPLOSION SURFACE WAVE
SIGNAL EXTRACTION, FB<0.1HZ, -4, AR-0, MA-15
Mon May 22 17:48:49 1995 SYNAPSE Science Center

Fig. 10.10b

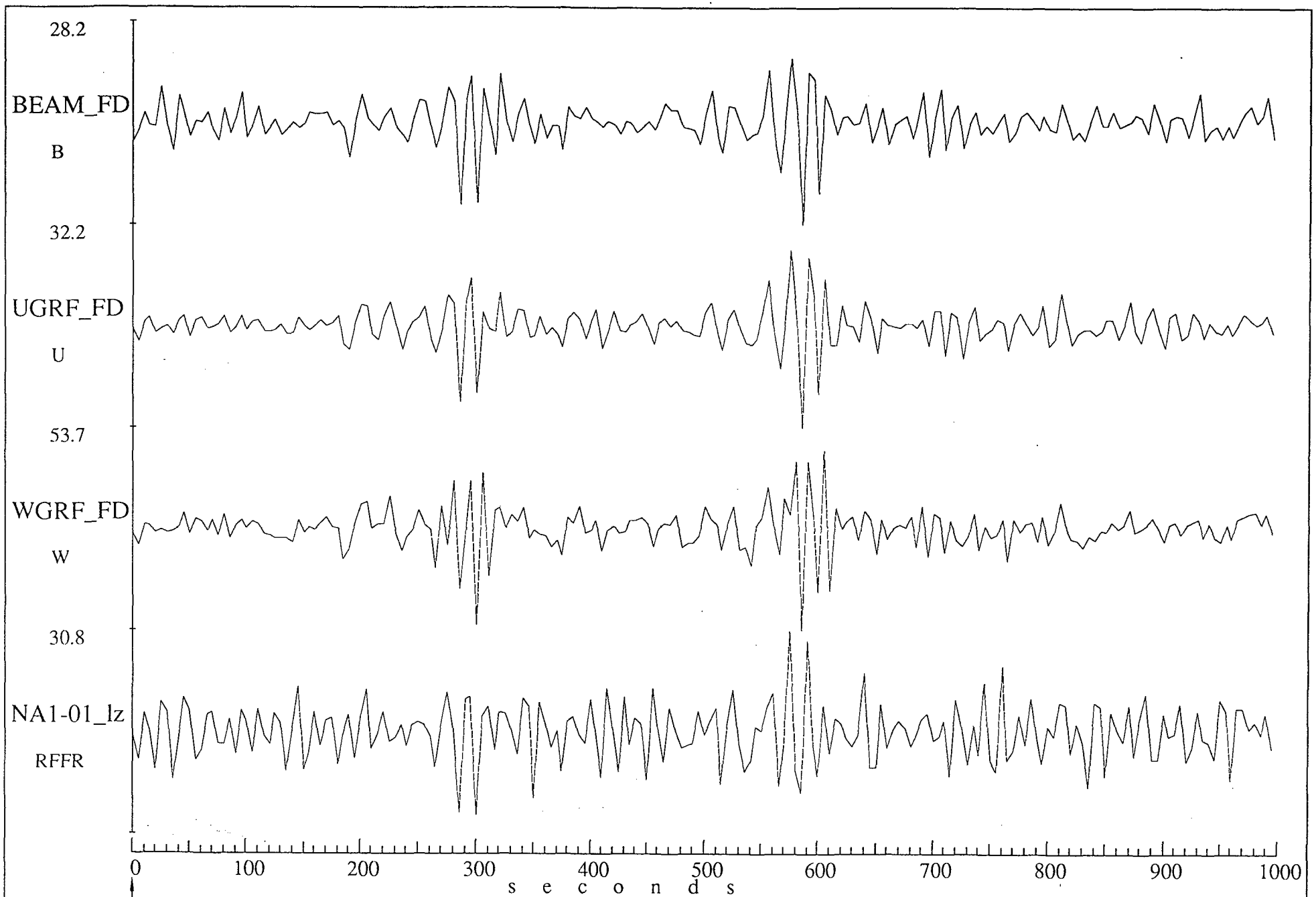

28.2
BEAM_FD
B
32.2
UGRF_FD
U
53.7
WGRF_FD
W
30.8
NA1-01_lz
RFFR
0
100
200
300
400
500
600
700
800
900
1000
s e c o n d s
Base time: 27.10.84 07.11.40. 0
NORSAR, EV2,z, NUCLEAR EXPLOSION SURFACE WAVE
SIGNAL EXTRACTION, FB<0.1HZ, -4, AR-0, MA-15
Mon May 22 17:50:20 1995 SYNAPSE Science Center

Fig. 10.10c

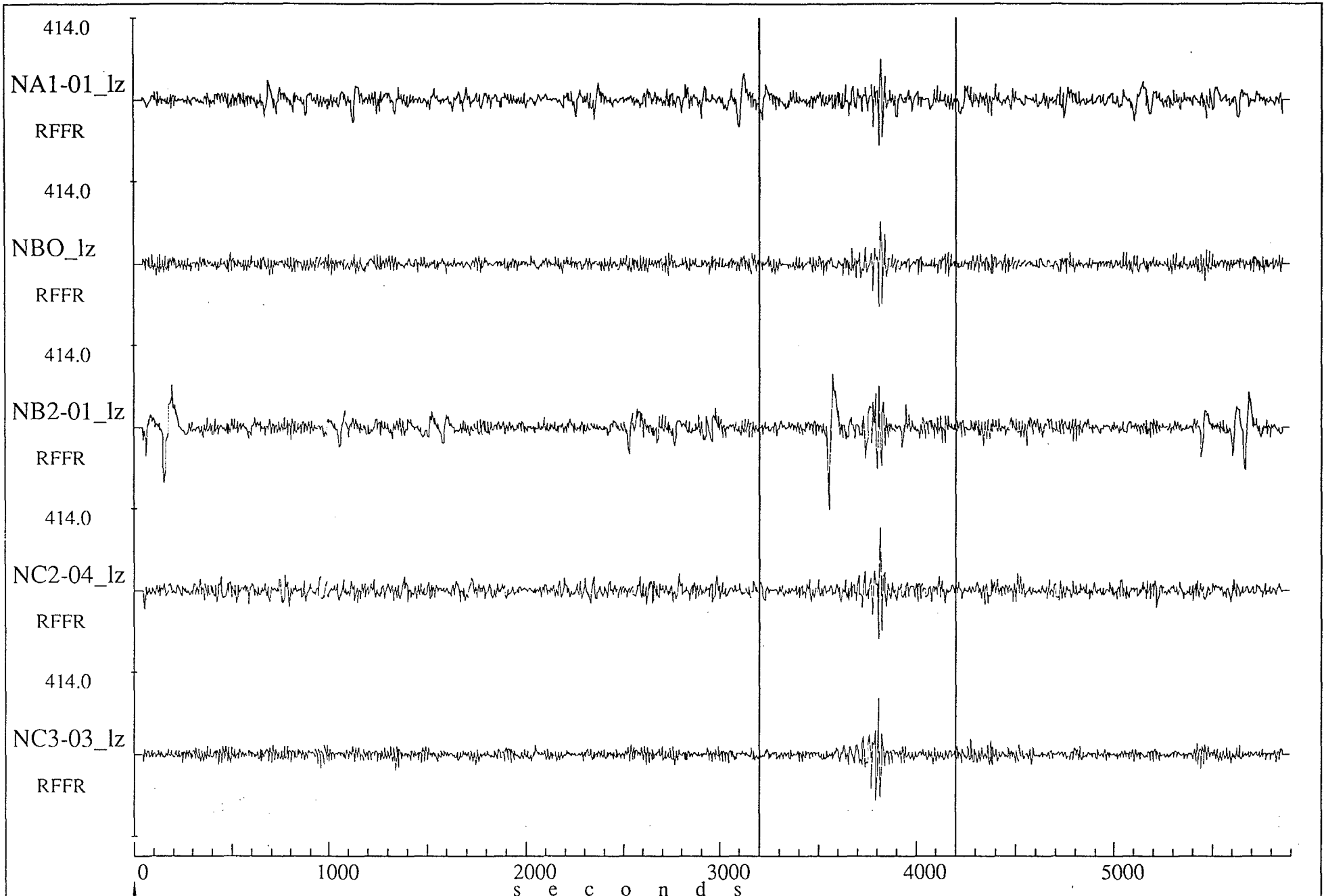

414.0
NA1-01_lz
RFFR
414.0
NBO_lz
RFFR
414.0
NB2-01_lz
RFFR
414.0
NC2-04_lz
RFFR
414.0
NC3-03_lz
RFFR
0
1000
2000
3000
4000
5000
seconds
Base time: 26.10.83 02.15.00. 0
NORSAR, EV3, NUCLEAR EXPLOSION SURFACE WAVES
WHOLE DATA INTERVAL, FB< 0.1 HZ
Mon May 22 17:52:30 1995 SYNAPSE Science Center

Fig. 10.11a

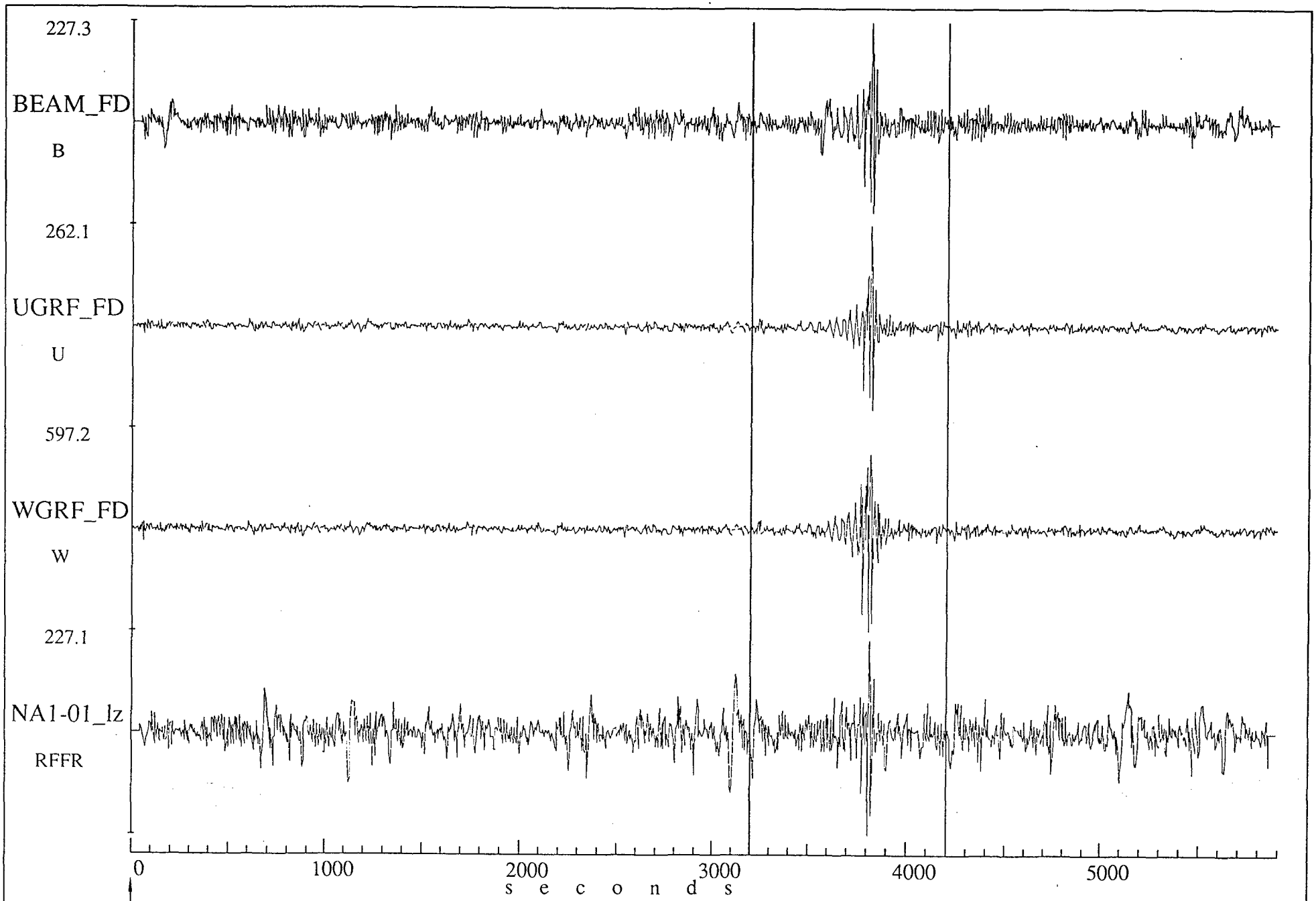

227.3
BEAM_FD
B
262.1
UGRF_FD
U
597.2
WGRF_FD
W
227.1
NA1-01_lz
RFFR
0
1000
2000
3000
4000
5000
seconds
Base time: 26.10.83 02.15.00. 0
NORSAR, EV3, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, -4, AR-0, MA-15
Mon May 22 17:53:11 1995 SYNAPSE Science Center

Fig. 10.11b

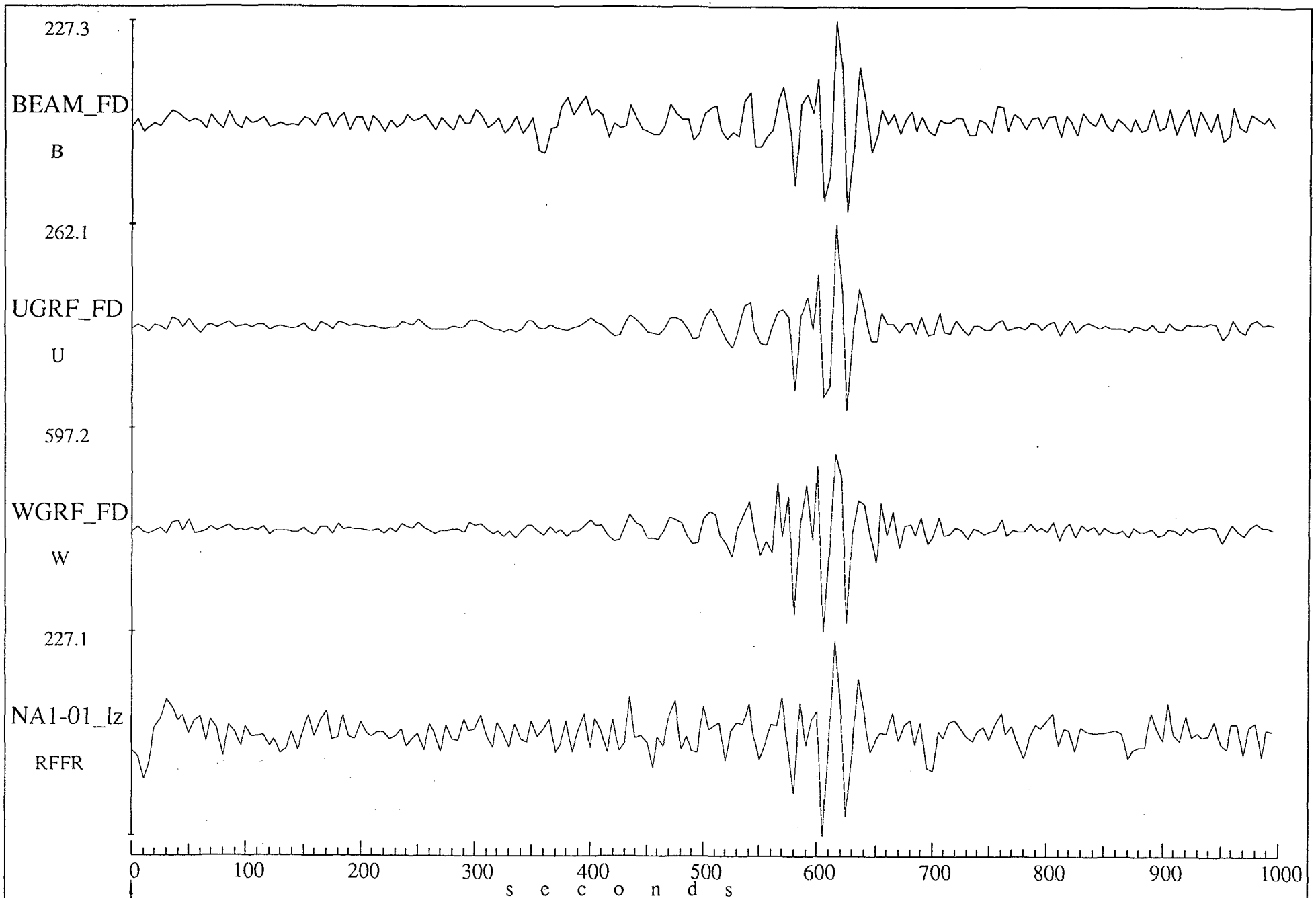

227.3
BEAM_FD
B
262.1
UGRF_FD
U
597.2
WGRF_FD
W
227.1
NA1-01_lz
RFFR
0
100
200
300
400
500
600
700
800
900
1000
s e c o n d s
Base time: 26.10.83 03.08.20. 0
NORSAR, EV3, NUCLEAR EXPLOSION SURFACE WAVES
SIGNAL EXTRACTION, FB<0.1HZ, -4, AR-0, MA-15
Mon May 22 17:55:00 1995 SYNAPSE Science Center

Fig. 10.11c

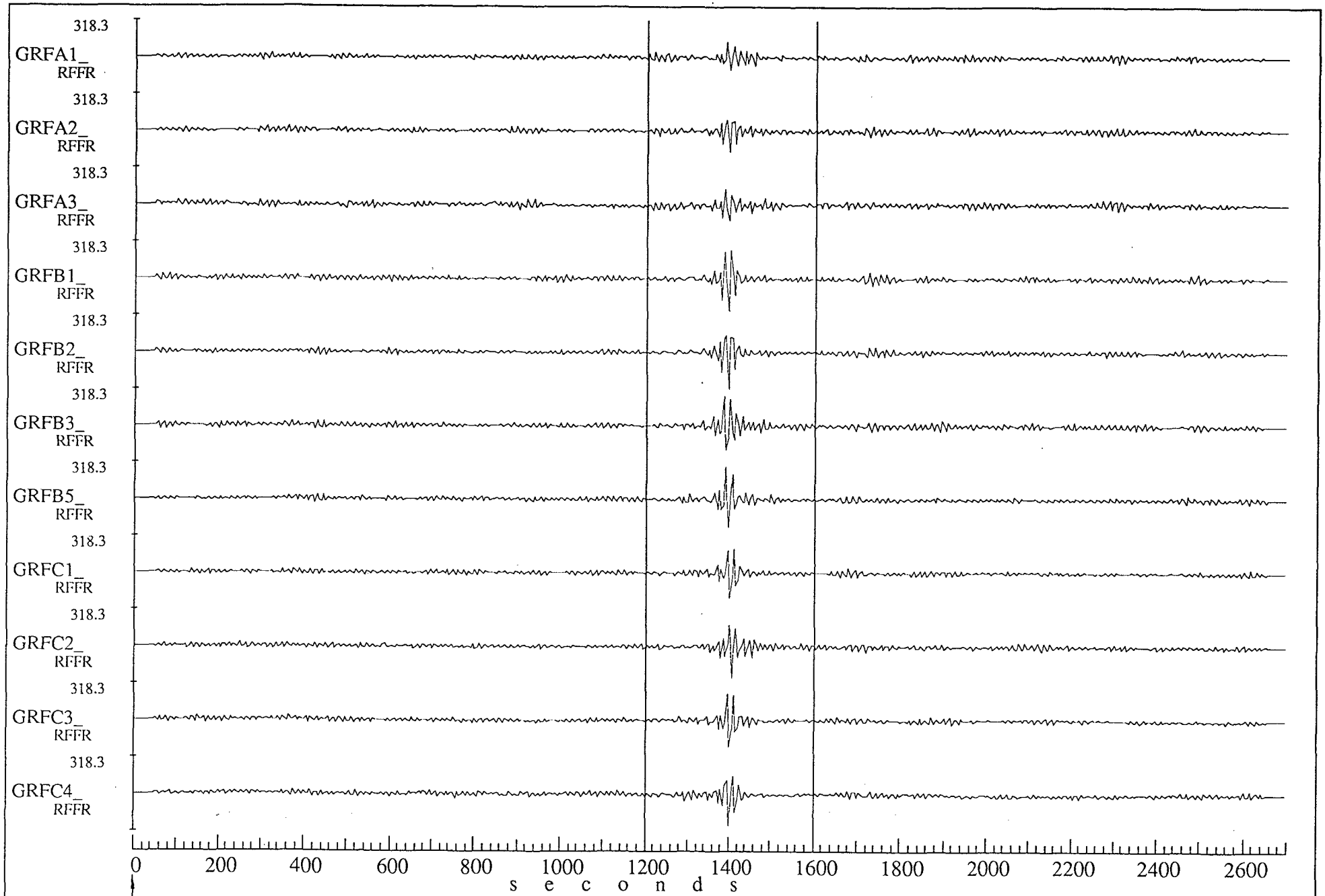

318.3
GRFA1_
RFFR
318.3
GRFA2_
RFFR
318.3
GRFA3_
RFFR
318.3
GRFB1_
RFFR
318.3
GRFB2_
RFFR
318.3
GRFB3_
RFFR
318.3
GRFB5_
RFFR
318.3
GRFC1_
RFFR
318.3
GRFC2_
RFFR
318.3
GRFC3_
RFFR
318.3
GRFC4_
RFFR
0 200 400 600 800 1000 1200 1400 1600 1800 2000 2200 2400 2600
s e c o n d s
Base time: 27.12.81 04.46.00. 0
GRAFENBERG, EV1z, NUCLEAR EXPLOSION SURFACE W
WHOLE DATA INTERVAL, FB< 0.1 HZ
Mon May 22 17:57:14 1995 SYNAPSE Science Center

Fig. 10.12a

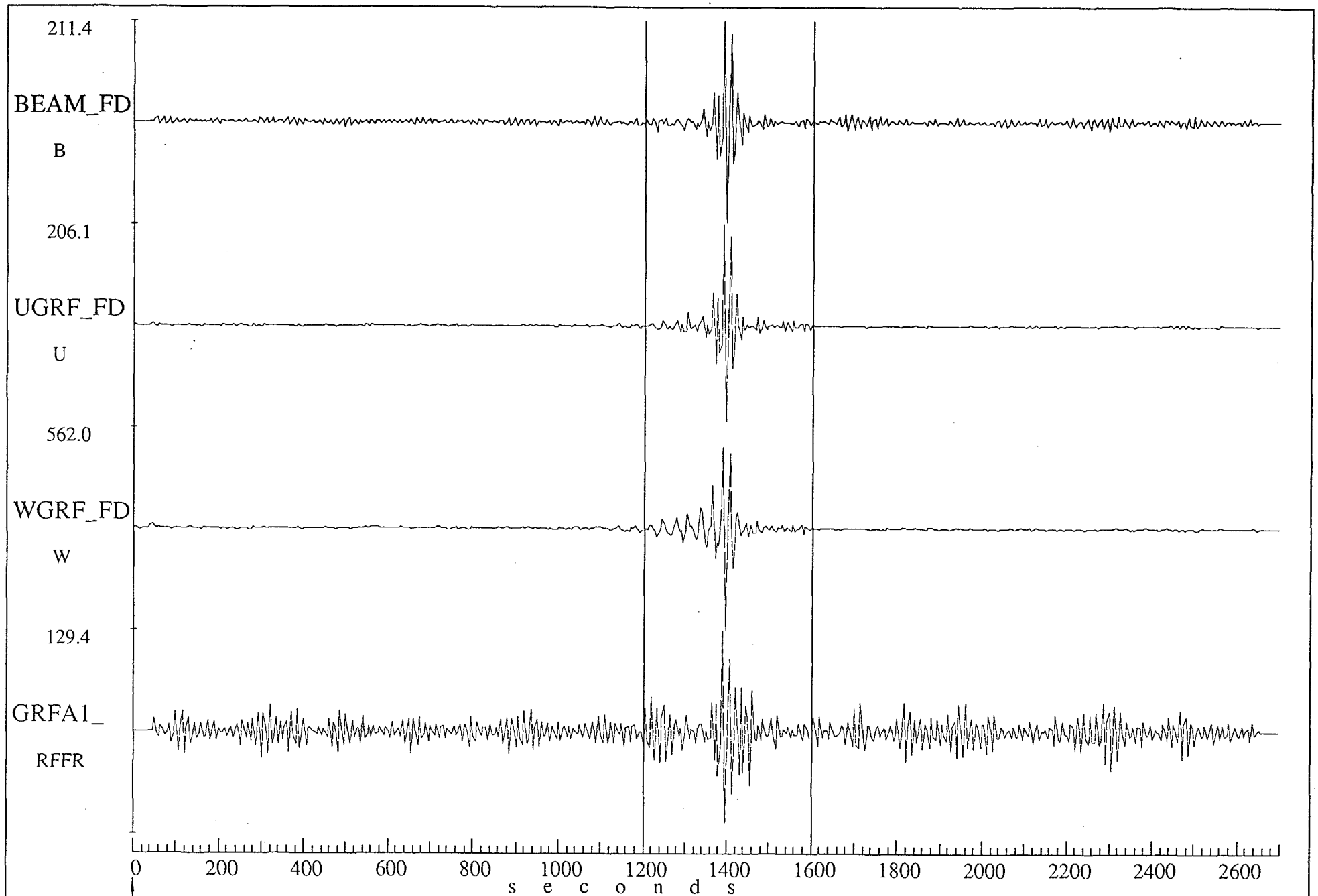

211.4
BEAM_FD
B
206.1
UGRF_FD
U
562.0
WGRF_FD
W
129.4
GRFA1_
RFFR
0
200
400
600
800
1000
1200
1400
1600
1800
2000
2200
2400
2600
seconds
Base time: 27.12.81 04.46.00. 0
GRAFENBERG, EV1z, NUCLEAR EXPLOSION SURFACE W
SIGNAL EXTRACTION, FB<0.1HZ, MA-15
Mon May 22 17:57:52 1995 SYNAPSE Science Center

Fig. 10.12b

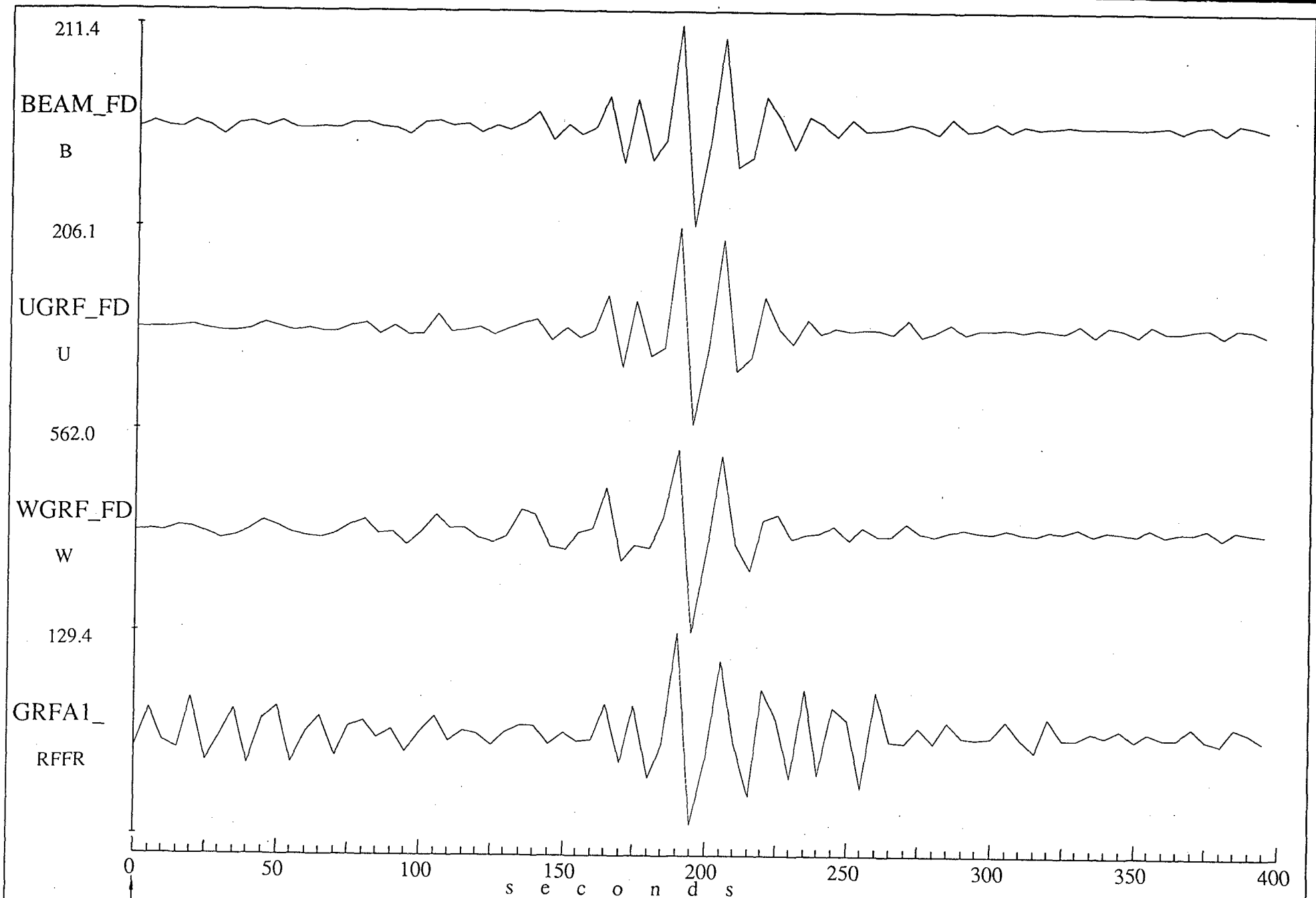
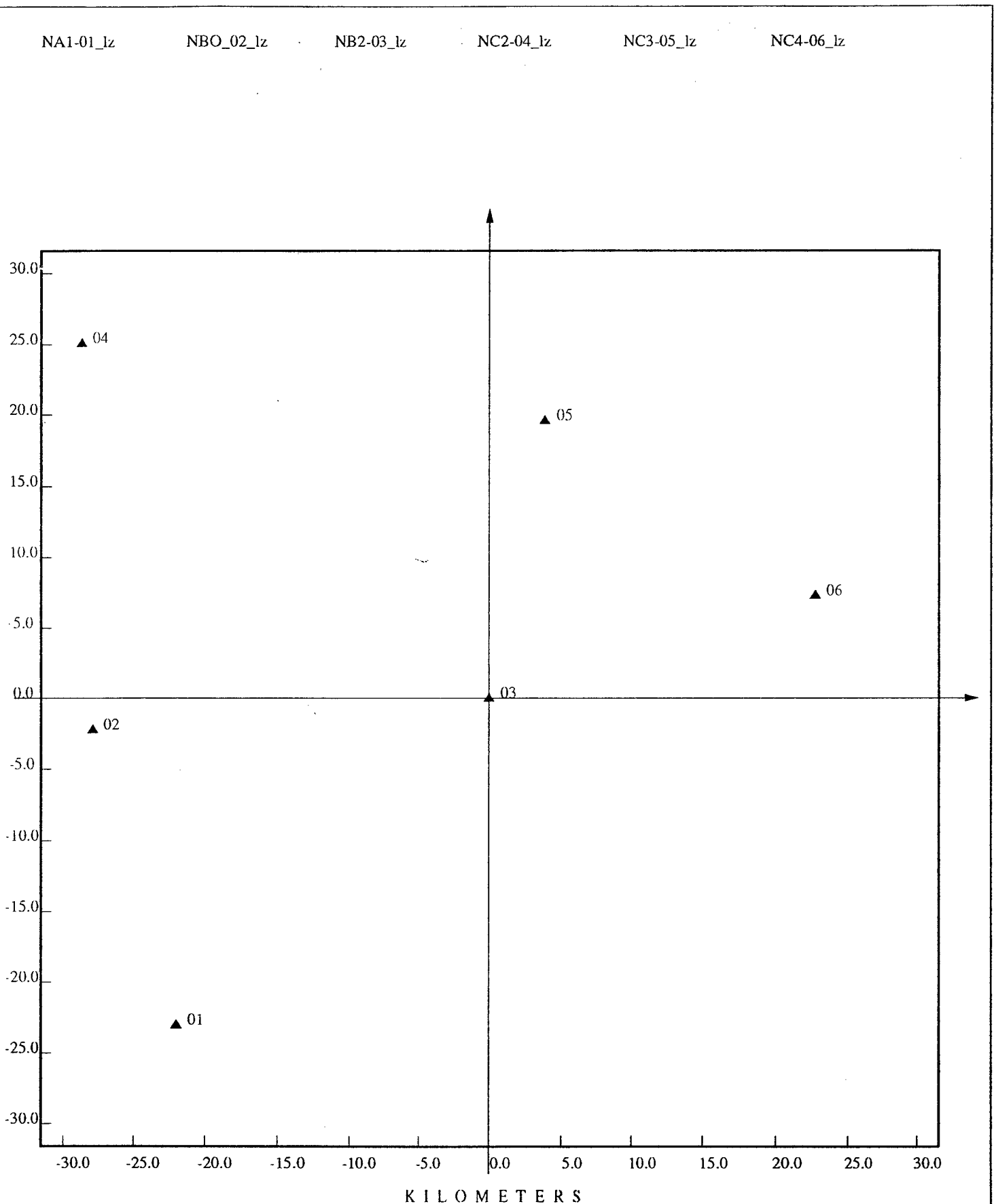

211.4
BEAM_FD
B
206.1
UGRF_FD
U
562.0
WGRF_FD
W
129.4
GRFA1_
RFFR
0
50
100
150
200
250
300
350
400
seconds
Base time: 27.12.81 05.06.00. 0
GRAFENBERG, EV1z, NUCLEAR EXPLOSION SURFACE W
SIGNAL EXTRACTION, FB<0.1HZ, MA-15
Mon May 22 17:58:02 1995 SYNAPSE Science Center

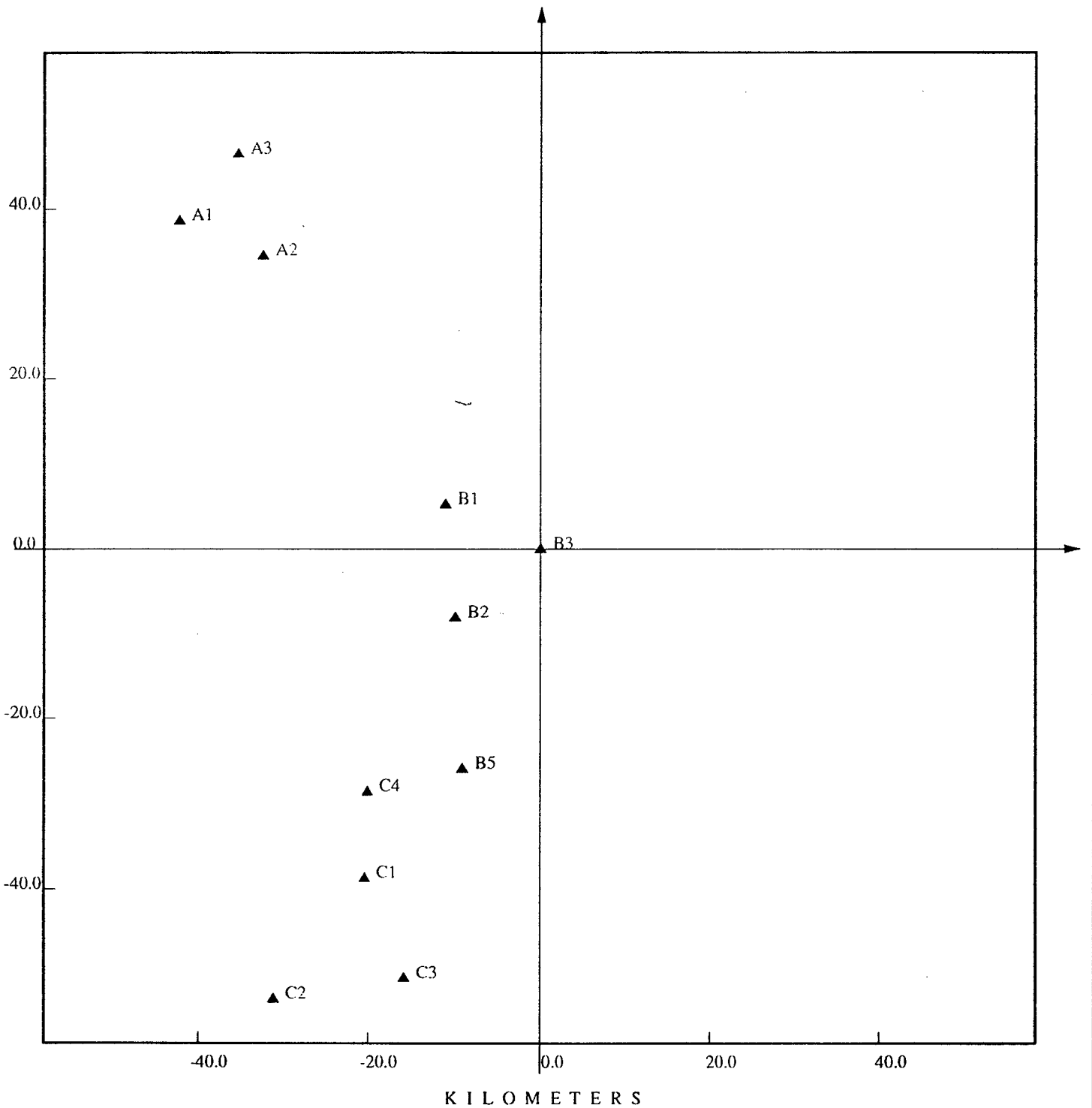
Fig. 10.12c



Configuration of seismic array - NORSAR_LPSA

Fig. 10.13 a

GRF_A1 GRF_A2 GRF_A3 GRF_B1 GRF_B2 GRF_B3
GRF_B5 GRF_C1 GRF_C2 GRF_C3 GRF_C4



Configuration of seismic array - GRAFENBERG_LPSA

Fig. 10.13b

```
SCRIPT 1

#SCRIPT: GEYOKCHA ARRAY DATA & TWO MODELS OF SURFACE WAVES
#script boris/albmsde.scr

#INITIAL VALUES OF BBV:
.int i=1, j=14, k=2
clearstack

#SIMULATION OF 12 CHANNELS SEISMOGRAM OF ONE SURFACE WAVE MODEL:
_bwarsim  boris/bwrsal.tmp
plot all

#GEYOKCHA ARRAY DATA READING:
readpack /detseis/seis/alex/data/alibek/Alibek_4.5fz.pk
winon 0 2500
cut

#DESIGNING OF THE ADDITIVE MIXTURE OF TWO SURFACE WAVES MODELS  AND
GEYOKCHA ARRAY DATA:
########################################
.lab1:
.if(i > 12) goto lab2
copy 1
scale 1 0
add 1 &j
shift 1 1700
add &k 1
#list
flush 1
copy 1
scale 1 0
add 1 &j
shift 1 2200
add &k 1
flush 1
.j = j + 1
.k = k + 1
.i = i + 1

.goto lab1
.lab2:
list
########################################

flush (13 - 24)
plot all

#OUTPUT DATA SAVING:
savepack /home/lap/boris/data/albsrf.pk

end
```

SCRIPT 2

```
# SCRIPT: SURFACE WAVES EXTRACTION FROM SEISMIC NOISE RECORDED BY
GEYOKCHA ARRAY

#script boris/alsmprps.scr

clearstack
readpack /home/lap/boris/data/albsrf.pk
#plot 1

# TRACE LOW PASS FILTERING FB < 0.45 HZ
filterS all 0.01 0.45 -1 -1  1
chanon 12
resample 12 10

list

# CHOOSING WINDOW, ADAPTATION AND FILTERING CHOOSED INTERVA
winon 0 1500
_marmamo boris/alspr.tmp -c

#SPATIAL SPECTRAL ANALYSIS
_spsp1 boris/spspalow1.tmp
#_tk1 -c

# CALCULATION OF THE ADAPTIVE GROUP FILTER FREQUENCY RESPONSE
_armagrf1 boris/alsach.tmp

# FILTERING OF THE WHOLE SEISMOGRAM
 winoff
_fpsfsa1 boris/alsaf.tmp
vertical r o 1750 ***
vertical r o 2250 ***

plot 6
hardcp  -Dh  b7b.ps
end
```

Script 3

```
#SCRIPT FOR SURFACE WAVE EXTRACTION FROM NOISE: NORSAR, EV1, n-
components

#script kush/sN1n.scr
clearstack
readpack /detseis/seis/alex/data/norsar/sN1h.pk
#plot all
winon 0. 7000.
cut all

# TRACE LOW PASS FILTERING < 0.1 HZ:
rmean all
#filtCresp  1 0.099 50 0.025 1
filterC all 0.099 50 0.025 5
keep 6
rmean all
footer 1 NORSAR, EV1,n, NUCLEAR EXPLOSION SURFACE WAVES
footer 2 LOW PASS FILTERED DATA FB< 0.1 HZ
#plot all -y

# CHOOSING WINDOW, ADAPTATION AND FILTERING CHOOSED INTERVAL:
#winon 0. 3000.
zwinon all 4000 800
chanon 6
vertical r o 4000
vertical r o 4800
#plot all -y
_marmamo kush/asN1v.inp
_armagrf1 kush/rsN1v.inp
_fpsfsa1
footer 1 NORSAR, EV1,n, NUCLEAR EXPLOSION SURFACE WAVES
footer 2 ADAPTATION INTERVAL, FB<0.1HZ, MA-15
#plot 4
flush 3
chanoff
zwinoff
#winoff

#  FILTERING WHOLE TRACE
footer 1 NORSAR, EV1,n, NUCLEAR EXPLOSION SURFACE WAVES
footer 2 WHOLE DATA INTERVAL, FB< 0.1 HZ
plot 6
_fpsfsa1
footer 1 NORSAR, EV1, NUCLEAR EXPLOSION SURFACE WAVES
footer 2 SIGNAL EXTRACTION, FB<0.1HZ, MA-15
plot 4 -y
winon 4000 800
winoff
end
#END OF SCRIPT
```

11. Statistical procedures of seismic data analysis

11.1 Program "marmamo": multidimensional ARMA modelling of seismic array data

Example of input fail

```
***PROGRAM MARMAMO: MULTIDIMENSIONAL TIME SERIES ARMA MODELING***
FILTERING MODE: 1- LP FILT.+ ARMA-MOD.; 0 - ONLY LP FILT.; -1 ONLY ARMA-MOD.
-1
LOW PASS FILTER CUT FREQUENCY (HZ)
5.
LENGTH OF LP FILTER IMPULSE RESPONSE (ONE SIDE)
31
PARAMETER OF FREQUENCY RESPONSE SLOPE
0.025
RESAMPLING FACTOR
4
NAME OF FILE FOR ARRAY SENSOR COORDINATES
data/noress.crd
ADAPTATION MODE: +/-1 -AR_WF; +/-2 -AR_LD; -3 -MA; -4 -ARMA
-4
ORDER OF MULTIDIMENSIONAL AUTOREGRESSIVE MODEL
10
ORDER OF MULTIDIMENSIONAL MOVING-AVERAGE MODEL
0
REGULARIZATOR VALUE FOR MATRIX AUTOCOVARIANCE FUNCTION
0.0001
STACK CHANNELS TO BE PROCESSED (examples: 10, (2-5), (1,3,4,8,9), all)
all
```

DESCRIPTION OF PROGRAM "MARMAMO"

The program calculates parameters of autoregressive-moving average (ARMA) model for multidimensional (vector) random time series. ARMA model of vector time series x(t)=(x(1,t),x(2,t),...,x(M,t))*, t=1,...,N, is a new random vector time series y(t)=(y(1,t),...,y(M,t))* which satisfies a following finite difference equation:

$$(1)\quad y(t) = \mathrm{SUM}[l=1,p]\{A(l)y(t-l)\} + \mathrm{SUM}[l=0,q]\{B(l)e(t-l)\},\quad t=1,..,N\ ,$$

where: e(k)=(e(1,k),...e(m,k))*, k=1,2,...,N, is a zero-mean Gaussian vector random time series with zero correlation between e(l,t) for different l,t and e(j,k) variances equal 1 for all l,t; A(l), l=1,p, and B(l), l=0,q, are (M x M) matrices: AR and MA matrix parameters of time series x(t). These parameters are evaluated to provide a fitness of statistical characteristics of model y(t)

with the same characteristics of time series x(t). The matrices A(l), l=1,p and B(l), l=1,q are the output data of the program and are saved to disc files with specified names.

The particular cases of ARMA modelling are autoregressive (AR) modelling and moving average (MA) modelling. In the first case matrices B(l)=0 for l=1,q, and B(0)=/=0. In the second case A(l)=0 for l=1,p. Program has several calculation modes providing different particular cases of ARMA modelling and different methods for calculation of AR and MA matrix parameters.

The program incorporates subroutine for LOW-PASS filtering of input components (channels) x(l,t), l=1,M by Chebyshev filter (the same filter is applied to all channels). A frequency response of the filter is calculated inside the program for given parameters: cut frequency, frequency response lope and resampling factor. Preliminary LOW-PASS filtering and resampling improves a fitness of a time series by a multidimensional ARMA model in a case where a high frequency part of data spectrum (starting from some cut frequency) contains much less power then the low frequency part. This case is a common for applications in seismology. The resampling factor after filtering have to be chosen to provide a new Nyquist frequency of data to be closest (but greater) then filter cut frequency.

The input vector time series x(t) are read into the program from the SNDA stack. An arbitrary sequence of channels can be chosen for consequent processing.

DESCRIPTION OF INPUT FILE PARAMETERS

FILTERING MODE [IFSW]:
Provides different modes for using program LP filtering facility.
If IFSW=1 then calculation of Chebyshev filter frequency response and data LP filtering and resampling precedes ARMA modelling.
If IFSW=-1 then ARMA modelling is performed without preceding data LP filtering and resampling.
If IFSW=0 then program performs only data LP filtering without ARMA modelling.

In the cases IFSW=O and IFSW=1 then filtered data are saved in the multiplex form and binary format to a disc file with the name 'ssa/fchn.dat' in the directory 'snda/sun4/ssa'.

LOW PASS FILTER CUT FREQUENCY (HZ) [FCH]:
LP filter frequency response (FR) is calculated to have the first zero at the cut frequency assigned. A FR is almost flat from zero up to frequencies slightly less then cut frequency and has the steep slope. FCH value have to be less than a Nyquist frequency of a filter input data

LENGTH OF LP FILTER IMPULSE RESPONSE (ONE SIDE) [IRL]:
The filter used is a zero-phase filter with symmetric impulse response (IR). IRL value is the number of non zero right side coefficients of the IR. The left side coefficients are the same and are ordered symmetrically. The filter quality depends on a IRL value: an increase of IRL yields a FR flatness in the filter pass band, a steepness of FR slope toward cut frequency and suppression of frequency amplitudes outside the filter pass band. From the other side this enlarge the transition process during filtering and time of calculation of the filter output. Recommended values of IRL = 30 - 100.

PARAMETER OF FREQUENCY RESPONSE SLOPE [ALPHA]:
Parameter ALPHA defines the width of FR slope and suppression of frequency amplitudes outside the filter pass band. An increase of ALPHA gains these characteristics. The ALPHA value can be calculated as: ALPHA=DF/FNQ, where DF is a width of slope (in HZ), FNQ is a Nyquist frequency of a filter input data (HZ).

RESAMPLING FACTOR [K]:
The LP filtering is performed with resampling of filter output trace. The resampling factor K have to be chosen to provide output trace Nyquist frequency to be closest but still greater then LP filter cut frequency. LP filtering with resampling improves fitness of data by ARMA model and speed up the consequent data processing.

NAME OF FILE FOR ARRAY SENSOR COORDINATES [COOFILE]:
File for sensor array data coordinates have to have the standard expansion '.scrd' and contain an M rows, where M is a number of array sensors. Each row have to consist on LABEL - 12 symbol character string, X-coordinate, Y-coordinate and Z-coordinate of a sensor (in km) (the centre of coordinates

is supposed as a rule to coincide with location of one of the array sensors). The path to the file can also be provided.

ADAPTATION MODE: +/-1 -AR_WF; +/-2 -AR_LD; -3-MA; -4-ARMA [ISWIT]:
Several modes for ARMA model coefficients are provided in the program:

The modes: ISWIT=-1 and ISWIT=-2 imply the evaluation of AR model with computation of the convolution of matrix AR coefficients :

(1) $L(k) = SUM[l=1,p]\{A(l)Binv(0)A^*(l-k)\}$, $k=0,p$,

where Binv(0) - inverse matrix for B(0): B(0)Binv(0)=I, * is the sign of transposition. Matrices L(k), k=0,p are the output of the program in this modes and are saved in the file 'ssa/invs.cft'.

The modes: ISWIT=+1 and ISWIT=+2 imply the evaluation of AR model with computation of optimal group filter vector coefficients g(k)=L(k)e, where e=(1,...,1) is the M-dimensional unit vector, g*(k)=(g(1,K),...,g(M,k))
Additionally calculation of scalar coefficients u(k) = e*L(k)e of Z-polynomial and coefficients v(k), k=0,p of minimal phase factorisation of this polynomial are performed. These parameters provide the multichannel data Winner group filtering in time domain. Vectors g(k) and coefficients u(k), v(k), k=0,p are the outputs of the program in these modes and are saved in the files 'ssa/ogf.cft ' 'ssa/infr.cft' and 'ssa/arf.cft', accordingly.

For ISWIT=+/-2 all matrix AR parameters: A(1),...,A(p) and B(0) are calculated with the help of multidimensional recursive Levinson-Durbin procedure [ ]. For ISWIT=+/-1 this procedure is used only for calculation of A(k), k=1,p. Matrix B(0) is estimated as covariance matrix of a multidimensional whitening filter output y(t):

(2) $B(0) = (1/N)\ SUM[t=1,N]\{y(t)y^*(t)\}$,
where
(3) $y(t) = SUM[k=1,p]\{A(k)x(t-k)\}$

The mode ISWIT=-3 implies estimation of a multidimensional MA model of data: e.g. A(1)=...=A(p)=0,

(4) $B(k) = (1/N)\ SUM[t=1,N]\{x(t)x^*(t-k)\}$, $k=0,...,q$.

Matrices B(k), k=0,q, are the output of the program in this mode and are saved in the file 'ssa/arm.cft'.

The mode ISWIT=-4 implies estimation of a multidimensional ARMA model: A(l), l=1,p are evaluated by multidimensional Levinson-Durbin procedure, B(k), k=1,q are estimated with the help of multidimensional whitening filter (3):

(5) $B(k) = (1/N) \sum_{t=1}^{N} \{y(t)t^*(t-k)\}, \quad k=0,...,q.$

Matrices A(k), k=1,p, B(k), k=0,q are the output of the program in this mode and are saved in the file 'ssa/mcov.fun' and 'ssa/whncov.fun', accordingly.

ORDER OF MULTIDIMENSIONAL AUTOREGRESSIVE MODEL [IP],
ORDER OF MULTIDIMENSIONAL MOVING-AVERAGE MODEL [IQ]:
These parameters must have the values not exceeding the maximum allocated in the program (in current version: 15). If IP=0 mode ISWIT=-3 is automatically assigned to save computation time. If IQ=0, this is preferable to assign ISWIT=-4 (if there is no needs for calculation of time domain filter parameters). This provides much faster calculation of AR matrix parameters as compared with ISWIT=-1 and ISWIT=-2, modes.

REGULARIZATOR VALUE FOR MATRIX AUTOCOVARIANCE FUNCTION [REG]:
This value is added to diagonal elements of multichannel data covariance matrix B(0) with the purpose to avoid unstability while calculating multidimensional ARMA model by Levinson-Durbin method. The unstability can occur if data are records of signal or noise close to coherent one. Recommended value of REG is in the limits 0.001 - 0.000.1. If a value of residual matrix determinant after Levinson-Durbin procedure (L-D RESID.DET.) is less than 1.E-30 a user is recommended to increase the REG value.

STACK CHANNELS TO BE PROCESSED
(examples: 10, (2-5), (1,3,4,8,9), all):
Symbols to be introduced here have to follow the rules provided in SNDA description and Stack Commands Help. A special procedure is provided in the program for extraction of sensor coordinates (from file COOFILE) for a subarray that corresponds to array channels ordered to be read from the SNDA stack for processing.

11.2 Program "armagrf": synthesis of vector frequency responses for adaptive and spatial rejecting group filters

Example of input file

```
***PROGRAM ARMAGRF: SYNTHESIS OF FREQUENCY RESPONSES FOR GROUP
FILTERS***
MODE OF CALCULATION: 0 - INVERSE SPECTRUM;  ADAPT.FLTRS;2 - REJECT.FLTR; 3 -
ALL FLTRS.; 4 - ONLY BEAM                                   (MCALC)
3
INITIAL AND FINAL POINTS FOR SCANNING AT AZIMUTH (DEGR) (XMN,XMX)
32.9  32.9
INITIAL AND FINAL POINTS FOR SCANNING AT APPARENT VELOCITY (SEC/KM)
(YMN,YMX)
10.4   10.4
INCREMENTS FOR SCANNING AT AZIMUTH, (DEGR) & APPARENT VRLOCITY (SEC/KM)
(DX,DY)
0.1    0.1
LOW & HIGH FREQUENCIES OF FILTER FREQUENCY RANGE (HZ) (FLH,FHH)
0.   5.
NUMBER OF FREQUENCIES                                       (NF)
128
AZIMUTH AND VELOSITY OF WAVE FOR REJECTING  ( for modes 2 & 3 ) (AZREJ,VREJ)
101.4   14.8
NUMBER OF ARRAY SENSORS  ( for modes 2 & 4 ) (L)
25
DATA SAMPLING INTERVAL   ( for modes 2 & 4 ) (DT)
0.1
FILE NAME FOR ARRAY COORDINATES  ( for modes 2 & 4 ) (CCOOFILE)
data/noress.crd
```

DESCRIPTION OF THE PROGRAM "ARMAGRF"

The program calculates vector frequency responses for 4 group filters at assigned frequencies and steering directions. Group filters being calculated are:

1) Adaptive optimal (Wiener) group filter (OGF) with the vector frequency response

$$(1) \qquad ro(f) = [h^*(f.p)\, Finv(f)]/[h^*(f,p)Finv(f)h(f,p)] ,$$

where $h(f,p) = (\exp\{-2\pi i f r'(j) ps\},\ j=1,...,m)$, $r(j)=(rx(j),ry(j))$ is X-Y coordinates of array sensors; $p=(psx,psy)$ is apparent slowness vector of plane wave in the given steering direction; Finv(f) is m x m inverse matrix

power spectral density (IMPSD) of array noise; * is the sign of Hermitian conjugation.

2) Adaptive noise whitening group filter (AWGF) with the vector frequency response

(2) rw(f) = [h*(f.p) Finv(f)]/SQRT[h*(f,p)Finv(f)h(f,p)] ;

3) Spatial rejecting group filter (SRGF) with the vector frequency response

(3) rr(f) = [h*(f.p) B(f,pr)]/[h*(f,p)B(f,pr)h(f,p)] ;

where B(f,pr) = [I - qq*/(q*q)], q(f,pr) = (exp{-2pifr'(j)pr}, j=1,...,m), pr=(prx,pry) is apparent slowness vector of an interfering plane wave to be rejected.

4) Conventional beam group filter (BGF)

(4) rb(f) = h*(f,p)/(h*(f,p)h(f,p)).

The IMPSD Finv(f) for AOGF and WGF is calculated in the program using matrix coefficients of ARMA-model for noise array record. These coefficients are provided by the program "marmamo" (which have to be run before the "armagrf" program) and are read from disc files with some standard names (see Description of program "marmamo"). Parameters used by the program "marmamo" during the adaptation are transferred to the program "armagrf" via the file 'ssa/invs.par'. The same 4 adaptation modes as for the "marmamo" are available in the "armagrf" and are controlled by a parameter MODSWIT which may have the following values: -1, -2, -3 -4.

For MODSWIT = -1 OR -2 Finv(f) in eq.1 and eq.2 is calculated as

(5) Finv(f) = SUM(j=-p,p)[L(k) exp{-i2(pi)k(f/fs)}].

where fs - is a data sampling frequency, L(k) - are matrices which are read from the file 'ssa/invs.cft'. These are convolutions of array noise matrix AR-coefficients (see Description of the "marmamo", eq.(1)).

For MODSWIT = -3 Finv(f) is calculated using equations

(6) Finv(f) = INV[F(f)], F(f) = SUM(j=-q,q)[C(k)w(k) exp{-i2(pi)k(f/fs)}]

where INV[U] is inverse matrix for U; w(k)=w(-k), w(k)= 1-k/(q-1) for k>0, are
Bartlett time window coefficients; C(k) = C'(-k) (' is the sign of transposition) are matrix covariances of array noise with lags k=0,q. They are read from the file 'ssa/mcov.fun'. These covariances are calculated as following

(7) $C(k)=(1/N)\ SUM(t=1,...,N-k)[x(t)x'(t+k)],\ k=1,...,q$

where x(t), t=1,N is a noise time series used for the adaptation of group filters.

For MODSWIT = -4 Finv(f) is calculated using equations

(8) $Finv(f) = P(f)Qinv(f)P^*(f),\ P(f) = SUM(j=0,p)[A(k)exp\{-i2(pi)k(f/fs)\}],$

$Qinv(f) = INV[Q(f)],\ Q(f) = SUM(j=-q,q)[Bw(k)w(k)exp\{-i2(pi)k(f/fs)\}],$

where A(k), k=0,...,p are the AR model matrix coefficients for array noise. They are read from the file 'ssa/arm.cft'; BW(k), k=1,...,q, are the matrix covariances of the whitened noise multiple time series (see Description of the "marmamo" program). They are read from the file 'ssa/whncov.fun'.

DESCRIPTION OF INPUT FILE PARAMETERS

MODE OF CALCULATION: 0 - INVERSE SPECTRUM; 1 - ADAPT.FLTRS; 2 - REJECT.FLTR; 3 - ALL FLTRS.; 4 - ONLY BEAM (MCALC) .
This parameter controls the type of program output:
MCALC=0, then only the estimate of IMPSD of array data is calculated using one of the AR-modelling methods (the latter is selected by the parameter MODSWIT). The IMPSD computation is made on the basis of matrix ARMA-model coefficients which are read from the disc files (see above). The set of the IMPSD matrices for equidistant frequency grid are saved at the disc file with the name 'inspec.mtr'.
If MCALC=1, then program calculates the vector frequency responses (VFR) for the AOGF, AWGF and BGF; if MCALC=2 - then for the SRGF and BGF; if MCALC=3 - then for all the 4 group filters; and if MCALC=4 - the program calculates VFR only for the BGF.

INITIAL AND FINAL POINTS FOR SCANNING AT AZIMUTH (DEGR) [XMN,XMX]
INITIAL AND FINAL POINTS FOR SCANNING AT APPARENT VELOCITY (SEC/KM) [YMN,YMX]
INCREMENTS FOR SCANNING AT AZIMUTH, (DEGR) & APPARENT VELOCITY (SEC/KM) [DX,DY]
The program can calculate a set of group filter corresponding some grid at the plane: (azimuth), i.e. for some fan of group filter steering directions. Parameters XMN, XMX define the interval of this grid in azimuths, YMN, YMX - in apparent slownesses. Parameter DX defines the grid equidistant steps through the azimuth axis, DY - the same through the ap.velosity axis.

LOW & HIGH FREQUENCIES OF FILTER FREQUENCY RANGE (HZ) [FLH,FHH]
NUMBER OF FREQUENCIES [NF]
These parameters control the equidistant grid of frequencies for which the group filter VFR values are calculated. Parameters FLH, FHH define the frequency band, parameter NF - the number of equidistant frequencies inside this band.

AZIMUTH AND VELOSITY OF WAVE FOR REJECTING (for modes 2 & 3) [AZREJ,VREJ]
This parameters define an arrival direction of interfering wave which have to be suppressed by the spatial rejection group filter. The AZREJ, VREJ parameters control the calculating the filter VFR in the program. These parameters are meaningful only for MCALC switcher values equal 2 & 3. For the other it's parameters AZREJ, VREJ are not used in the computations and can have any values.

NUMBER OF ARRAY SENSORS (for modes 2 & 4) [L]
DATA SAMPLING INTERVAL (for modes 2 & 4) [DT]
FILE NAME FOR ARRAY COORDINATES (for modes 2 & 4)
These parameter assignment is essential in calculation modes: MCALC={2, 4}, where the program "armagrf" may be fun independently from the program "marmamo". In the modes MCALC= {0, 1, 3} any performance of the "armagrf" must be proceeded by running the "marmamo". So in this modes the "armagrf" gets values of all these parameters from the disc file 'ssa/ivs'.par' which serves as a parameter interface between the "marmamo"

and "armagrf" programs. In the modes {0, 1, 3} the parameters being discussed may have arbitrary values.

The output of the program is saved in the disc files with the standard names:
Estimate of inverse matrix spectral density (if MCALC=0) - in the file 'ssa/invsp.par'; parameters connected with this IMPSD estimate: ARMA-model, frequency band, number of frequencies, number of sensors, sensor coordinates and so on - in the file 'ssa/insm.par'; calculated vector frequency response (VFR) for AOGF - in the file 'ssa/aogf.par'; VFR for AWGRF - in the file 'ssa/aogfwh.fr'; VFR for RJGF - in the file 'ssa/srjf.fr'; VFR for BGF - in the file 'ssa/bmf.fr'. Parameters of group filter vector frequency responses: frequency band, number of frequencies, set of steering directions, number of array sensors, array sensor coordinates, ARMA-model for IMPSD estimate and so on, are saved in the disc file 'ssa/grf.par'.

11.3 Program "fpsfsa": group filtering in frequency domain

Example of input-file

```
****PROGRAM "FPSFSA":GROUP FILTERING IN FREQUENCY DOMAIN****
INPUT/OUTPUT: 1 - FROM/TO THE SNDA STACK:, 0 - FROM/TO DISC FILES
0
OUTPUT OF FILTERED TRACES: 0 - IN THE FREQUENCY DOMAIN, 1 - IN THE TIME
DOMAIN, -1 - WITHOUT TRACE OUTPUT
1
TIME SHIFT OF TRACES (IN POINTS) IN THE TIME DOMAIN
0
```

DESCRIPTION OF PROGRAM 'FPSFS

The program performs a group filtering of array data in the frequency domain. Data for treating are entered from the SNDA stack or from the disc files 'fchn.dat', 'fchn.par', produced by the "marmamo" program. Group filter vector frequency responses (VFR) calculated by the program "armagrf" are read from the following disc files:
(VFR) for AOGF - in the file 'ssa/aogf.par'; VFR for AWGRF - in the file 'ssa/aogfwh.fr'; VFR for SRGF - in the file 'ssa/srjf.fr'; VFR for BGF - in the file 'ssa/bmf.fr'. Parameters of group filter VFR's, such as frequency band, number of frequencies, set of steering directions, number of array

sensors, array sensor coordinates, ARMA-model for IMPSD estimate and so on, are read from the disc file 'ssa/grf.par'

In depend on a value of the parameter MCALC (delivered from the "armagrf" program via the file ssa/grf.par') the "fpsfsa" program calculates output traces after the different types of group filters: for MCALC=1 - after the AOGF, AWGF and BGF; for MCALC=2 - after the SRGF and BGF; for MCALC=3 - for all the group filters involved; for MCAL=4 - only for the BGF. (the abbreviations for fitters are the same as for "armagrf" program). The output traces are placed sequentially in the beginning of SNDA stack or saved into disc files with NORSAR ASCII format. The names of output files are: 'aogf.out', 'awgf.out', 'srjf.out' and 'bmf.out'. The output traces can be produced by the program either in the time or in the frequency domain.

Upon to request made in the program "armagrf" the program "fpsfs can provide the 'fan' of group filtering, corresponding to some grid of steering directions. Besides transforming the filtered traces to the time domain (this option can be switched off) the program calculates the averaged power for every trace produced. These values are printed in the screen and stored in the files 'aogr.map', 'bmf.map', 'awgf.map', 'srjf.map'. These option allows to evaluate the effectiveness of group filtering application for coherent noise suppressing and to calculate the power map for signal waves arriving to the array from different directions.

The peculiarity of the program is that the frequency domain group filtering does not depend on the relations between the set of frequencies for which the VFR were calculated in the program "armagrf" and the set of Discrete Fourier Transform frequencies for the data being treated. The only restriction is that both the sets have to belong to the same frequency band. This allows to perform group filtering the data at any long time intervals with the same VFR. This facility is provided because the program interpolates the VFR values to any frequencies between two consequent frequencies for which VFR have been previously calculated

DESCRIPTION OF INPUT FILE PARAMETERS

INPUT/OUTPUT: 1 - FROM/TO THE SNDA STACK:, 0 - FROM/TO DISC FILES
This switch allows to run the program outside the SNDA framework. The standard input and output file names are used in the "0" mode.
OUTPUT OF FILTERED TRACES: 0 - IN THE FREQUENCY DOMAIN, 1 - IN THE TIME DOMAIN, -1 - WITHOUT TRACE OUTPUT

This switch allows to get the different modes for program outputs; if it is equal -1, then only the maps for output trace power are produced and stored in the disc files.

TIME SHIFT OF TRACES (IN POINTS) IN TIME DOMAIN
This parameter provides the time shift of filtered traces with the purpose to compensate some inherent time shifts in other procedures implemented during data processing

11.4 Program "fkan": multimode F-K analysis

Example of input-file

```
**********PROGRAM FKAN: MULTIMODE F-K ANALYSIS***************
LOW FREQUENCY OF ANALISYS FR. BAND (HZ)                    (FLOWF)
3.
HIGH FREQUENCY OF FR. BAND FOR ANALISYS(HZ)                (FHIGF)
3.
STEP IN FREQUENCY (HZ)                                     (STEPF)
0.1
NUMBER OF POINTS IN F-K MAP                                (IGRIDP)
41
MAXIMAL SLOWNESS IN F-K MAP (SEC/KM)                       (SLOWM)
.4
START TIME (SEC FROM TIME SERIES START P. )                (STRT)
0.
END TIME (SEC FROM TIME SERIES START P.                    (STOP)
50.
REGULARIZATOR VALUE FOR MATRIX AUTOCOVAR. FUNCTION         (REG)
.0010
COVAR. FUNCTION LENGTH FOR ANALISYS                        (IP)
20
F-K ESTIMATION METHOD: L - LOW RESOLUTION; B - CAPON-BARTLETT; C -
CAPON-AR; M - MUSIC ; E - EIGEN VECTOR;                          (MODE)
E
SCALING OF MAP (LINEAR-LIN, LOGARITHMIC-LOG)          (ISCALE)
LIN
DIMENSION OF SIGNAL SPACE (FOR MUSIC AND EIGEN VECTOR METHOD)    (NSIG)
2
KIND OF MAP ( SLOWNESS OR VELOCITY)                        (MAPTYPE)
SLOW
FILE FOR COORDINATES OF ARRAY SENSORS                      (FILEC)
data/noress.cr
OUTPUT FILE FOR THE MAP                                    (FILEM)
data/fk.map
STACK CHANNELS TO BE PROCESSED (examples: 10, (2-5), (1,3,4,8,9), all)
all
```

DESCRIPTION OF PROGRAM "FKAN"

The frequency-spatial spectrum (F-K spectrum) of a homogeneous random seismic field z(r,t) being observed at the Earth surface is described by the following equation,

$$p(f,p) = INT[\exp\{-i\, p'r\}F(f,r)]dr \qquad (1)$$

where p=(px, py)' - is a apparent slowness vector of plain wave arriving to the Earth surface; f - is a frequency of wave oscillations; r = (x,y) is a radius-vector of the point (x,y) at the Earth surface; F(f,r) - is a power spectrum density of the random field equal Fourier transform of covariance function of the field; the integral is computed for entire X-Y plane; ' is the sign of transposition.

For seismic arrays as for discrete observation systems where sensors are located in finite number m of sites on the Earth surface all the information concerning F(f,r) is contained in a Hermit matrix power spectrum of the array signals:

$$F(f) = \begin{vmatrix} F11(f), F12(f), \ldots, F1m(f) \\ F21(f), F22(f), \ldots, F2m(f) \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ Fm1(f), Fm2(f), \ldots, Fmm(f) \end{vmatrix} \qquad (2)$$

This matrix consists of elements corresponding to autospectra and cross-spectra of signals in the array sensors.

In the program the following 5 methods are implemented for estimation of seismic field F-K spectrum based on seismic array data:

1. **Low resolution (LR) F-K spectrum estimate.**

This estimate is produced by the formula,

$$PL(f,p) = (1/m)E^*(p,f)F\hat{}(f)E(p,f), \qquad (3)$$

where E(p,f)=(e(1,p,f),e(2,p,f),...,e(m,p,f))' - is a steering vector with components; e(j,f)=exp[-i(p'r(j))]; r(j) is a vector of coordinates of j-th array

sensor; F^ is an estimation of matrix power spectrum of array signals; * denotes the Hermitian conjugation, ' denotes the transposition.

For estimation of matrix power spectrum F(f) the correlation method (Bartlet method) is used, including the computation of matrix correlation function, multiplying it (whitening) by some time window and then Fourie transforming. Eq(3) straightforwardly follows from eq.(1) if substitute integrating over entire the X-Y plain by summing through observational points r(j) where array sensors are located. This F-K spectrum estimate provides low resolution capabilities while estimating several plain waves arriving from different directions, especially in the case of small aperture arrays. The following methods were designed to guarantee the high resolution capabilities in the latter problem.

2 **Capon-Bartlett (CB) F-K spectrun estimate**.

This estimate put forward by Capon is

designed for seismic wave separation. It was derived by the maximum likelihood method for a problem of plane wave extraction from heterogeneous noise field. This estimate is given by the formula

$$PCB(f,k)= 1 / [E^*(f)\ F\hat{}inv(f)\ E(f)], \qquad (6)$$

F^inv(f) = [F^ (f)]**-1 denotes an inverse matrix for an estimate of matrix power spectrum eq.(1). The latter estimate is the same as for LR method.

3. **Capon autoregressive (CAR) F-K spectrun estimate**

This method is a modification of the previous estimation. The difference is that a different model of multidimensional time series being analysed is used: it is assumed that the time series is well approximated by the multidimensional autoregressive model. The inverse matrix power spectrum of this model is described by :

$$Far\ inv(f) = Q(f)\ G\ Q^*(f), \qquad (7)$$

where

$$Q(f) = SUM(j=0,p)[A(k)exp\{-i2(pi)k(f/fs)\}], \qquad (8)$$

A(k), k=1,..,p, are square matrixes of multidimensional AR model of the analysed time series; G is a covariance matrix of AR-model residual; fs - is a data sampling frequency.

The CAR F-K estimate PCR(f) is calculated in accordance with eq.(6) where the estimate F^ar inv(f) of inverse matrix power spectrum by eq(7) is

substituted instead the Bartlett estimate F^inv(f). The estimates A^(k) of multidimensional autoregressive model matrix coefficients are computed by solving the following system of Youl-Walker linear equations

$$\text{SUM}(k=1,...,p)[A^*(k)C(k-l)] = C(l),\ l=1,...,p,$$
$$G = C(0) - \text{SUM}(k=1,...,p)[A^*C(k)] \qquad (8)$$

where

$$C(k)=(1/N)\ \text{SUM}(t=1,...,N-k)[x(t)x'(t+k)],\ k=1,...,p$$

The computationally effective recurrent procedure is implemented which is a generalisation of the well known Levinson-Durbin procedure.

Two non-linear methods of F-K spectrum estimation (MUSIC and EV) are also represented in the program. This methods are based on singular value decomposition of matrix power spectrum of array data and analysis of its eigen vectors and eigen values. This decomposition has the form :

$$F(f) = \text{SUM}(k=1,...,m)[q(k)V(k)V^*(k)] \qquad (9)$$

where q(k),V(k) are eigen values and eigen vectors ordered in decreasing q(k) values.

4. **Multiple signal classification (MUSIC) F-K spectrun estimate.**
This estimate is computed from the formula:

$$Pmsc(p)=1 / E^*(f)\{\text{SUM}(k=ns+1,...,m)[V(k)V^*(k)]\}E(f), \qquad (10)$$

where ns is a dimension of the signal space, m-ns is a dimension of the noise space.

5. Eigen vector (EV) F-K spectrun estimate
This estimate is computed from the formula:

$$Pev(p)=1 / E^*(f)\{\text{SUM}(k=Ns+1,M)[(1/q(k))\ V(k)V^*(k)]\}E(f) \qquad (11)$$

DESCRIPTION OF INPUT FILE PARAMETERS

LOW FREQUENCY OF ANALISYS FR. BAND (HZ) [FLOW]
HIGH FREQUENCY OF ANALISYS FR. BAND (HZ) [FHIGH]
These parameters are law and high edges of a frequency band where the F-K spectrum is computed.

STEP IN FREQUENCY (HZ) [STEPF]
The program calculates the set of F-K spectrum maps for equidistant grid of frequencies with given steps within the band assigned. These maps are averaged then before plotting

NUMBER OF POINTS ALONG THE X & Y MAP COORINATES [IGRIDP]
MAXIMAL SLOWNESS IN F-K MAP (SEC/KM) [SLOWM]
These parameters assign the 2-dimensional rectangular grid of slowness vectors for which values of F-K spectrum are calculated in the program. The first parameter defines the number of the grid points (equal IGRIDP x IGRIDP), the second - the maximal grid slowness, the same along the X & Y axes in both (+/-) directions. It should be emphasised that the program calculates the map in the square with the centre placed into the point with px=0, py=0.

START TIME (SEC FROM TIME SERIES START P.) [STRT)]
END TIME (SEC FROM TIME SERIES START P.) [STOP]
These parameters assign a data time interval of the input multiple time series for which F-K estimate is calculated. Both the parameters are counted in sec from the beginning of time series.

REGULARIZATOR VALUE FOR MATRIX AUTOCOVAR. FUNCTION [REG]
When the signal matrix power spectrum is close to singular (the case of 'pure' coherent seismic field), it is necessary to apply the regulation procedure to improve the correctness of this matrix inversion. This may be done by setting a small (e.g 0.0001) value to REG parameter.

COVAR. FUNCTION LENGTH FOR ANALISYS [IP]

This parameter defines the statistical errors of the F-K spectrum estimate and its resolving power. It regulates the estimate smoothness over some frequency band around of frequency f for which this estimate is calculated. Increasing of IP decreases the smoothness i.e. increase the stochastic fluctuations of the F-K spectrum estimate over slowness vector plain. But a capability of the estimate to resolve distinct plane waves coming from different directions increases in this case. Theoretically, some compromise is achieved if IP is approximately equal 0.02 - 0.1 of the number of data points at the time interval chosen for analysis. If amount of array channels is large the left margin of this interval can be used.

F-K ESTIMATION METHOD: LOW RESOLUTION-L; CAPON-BARTLETT - B; CAPON-AR - C; MUSIC - M; EIGEN VECTOR - E; (MODE)
The parameter to select the method for estimation of F-K spectrum from those which is described above.

SCALING OF MAP (LINEAR-LIN, LOGARITHMIC-LOG) [ISCALE]
This parameter controls the representation of the F-K spectrum along the Z-coordinates. The Log and Linear scales may be set by the Log and LIN values of the parameter.

DIMENSION OF SIGNAL SPACE (FOR MUSIC AND EIGEN VECTOR METH.) [NSIG]
The resolution of MUSIC and EV methods is controlled by this parameter, which determines the dimension of signal space. If the amount of signal waves is known in advance, this value should be assigned to the dimension of signal space NSIG. However in most practical cases this number is unknown. For better reliability of signal waves separation it's useful to compute several maps with different values of this parameter.

KIND OF MAP (SLOWNESS OR VELOCITY) [MAPTYPE]
It's also possible to compute the map for F-K spectrum for velocity coordinates Vx, Vy instead of slowness coordinates. This option is controlled by this parameter which can take the values: SLOW and VEL.

FILE FOR COORDINATES OF ARRAY SENSORS [FILEC]
This string parameter assigns the name of file containing the labels and coordinates of array sensors which signals are to be processed.

OUTPUT FILE FOR THE MAP [FILEM]
This string parameter assigns the name of file in which the values of F-K spectrum estimate is to be placed.

STACK CHANNELS TO BE PROCESSED (examples: 10, (2-5), (1,3,4,8,9), all)
This string parameter assigns the channels of the SNDA stack to be read for F-K analysis. The labels of this channels have to be contained in the list of array sensor labels in the previous file.

The maximum values of numerical variables in the program depend on FORTRAN features and amount of computer memory available. The following maximal values are set now:

Max amount of array sensor	25
Max number of data points in multiple time series	4096
Max number of points in the F-K spectrum map	100x100
Max number of lags for autocorrelation function	30

The new maximal values of these variables can be set in the program C-header with the name "c_fkan".

The program result is 2-dimensional data array with the values of F-K spectrum estimate on the equidistant grid saved in the disc file. It used to be an input file for level-grid map plotting by the UNIX utility "contour" which is run together with the SUN-system raphic tool set. The program "fkan" creates the control input file for the "contour" utility and place it to the disc with the name 'plot/contour.inp'.This file comprises headers, footers and other raphic attributes for the map.

11.5 Program 'phasedet": detecting weak seismic phases in wavetrain by adaptive statistical detector

Example of input-file

```
****PROGRAM PHASEDET: DETECTING SEISMIC PHASES****
READ/WRITE DATA: 0 - FROM/TO SNDA STACK:, 1 - FROM/TO FILE
0
NAME OF FILE WITH TRACE FOR PROCESSING (for reading from file)
bmf.out
STACK CHANNEL (N) (for reading from stack)
(1)
START POINT FOR PROCESSING (SEC)
0.
LEHGTH OF INTERVAL FOR PROCESSING (SEC)
60.
ORDER OF AUTOREGRESSIVE MODEL FOR DETECTING
5
LENGTH OF MOVING WINDOW (IN SAMPLES)
30
DATA SAMPLING INTERVAL (SEC) (for reading from file)
0.1
```

DESCRIPTION OF PROGRAM "PHASEDET'

The program performs the adaptive statistically optimal detecting of weak wave phase signals inside a one-component seismogram wavetrain. In this variant of the program the adaptation for average noise characteristics are made using total data interval being treated. As the result of adaptation autoregressive model coefficients of the wavetrain are estimated and the wavetrain is processed by the whitening filter incorporating those coefficients. i.e. the trace is produced:

$$n(t) = SUM(k=0,...,p)[a(k)x(t-k)] , \qquad t = t0,...,t1, \ t1=t0+l,$$

where x(t) - is the trace being treated; t0 -s a start time for processing the trace; l - is the data time interval for processing; a(k), k=0,...,p, - are the autoregressive (AR) model coefficients calculated for x(t), t=t0,...,t0+l with Levinson-Durbin procedure.; p - is the order of AR model

The second stage of detecting procedure is the calculation of detector statistic values in a moving time window by following equation

Ds(T) = SUM(k=0,...,p){(SUM(t=0,...,w)[n(T-t)n(T)])**2}, T=t0+w,...,t1-w

The trace of Ds(T) values is the program output which is placed in the beginning of SNDA Stack or to an assigned output file. The adaptive statistically optimal detector implemented is the high sensitive one and responds on deviations of power and spectrum content of data inside the moving window in respect with the averaged power and spectrum of the total trace.

The peculiarity of the program is the capability to evaluate a detection threshold based on detector statistic fluctuations. A threshold value is calculated under condition to reveal an assigned amount (say, 2 or 3) of the most strong peaks in the detection statistic trace. The time moments of crossing the threshold by these peaks in up and down directions are delivered to the SNDA Black Board. This allows to arrange the automatic detection of an assigned amount of the most strong phases in a seismic wavetrain.

DESCRIPTION OF INPUT FILE PARAMETERS

READ/WRITE DATA: 0 - FROM/TO SNDA STACK:, 1 - FROM/TO FILE
This switch allows to run the program outside the SNDA framework

NAME OF FILE WITH TRACE FOR PROCESSING (for reading from file)
This name assigns the binary data file with single trace to be processed.

STACK CHANNEL (N) (for reading from stack)
This is the string for Stack channels assignment using SNDA notations

START POINT FOR PROCESSING (SEC)
The initial point of the data time interval to be processed.

LEHGTH OF INTERVAL FOR PROCESSING (SEC)
The length of the data time interval to be processed.

ORDER OF AUTOREGRESSIVE MODEL FOR DETECTING
This value is to be in the limits 3-7

LENGTH OF MOVING WINDOW (IN SAMPLES)

This value should correspond an expected duration of seismic phase being detected, but have to be at least ten time less then data time interval being processed

DATA SAMPLING INTERVAL (SEC) (for reading from file)
This is the self explained parameter

AMOUNT OF PEAKS ABOVE A THRESHOLD [PK]
This parameter defines a threshold value providing triggering the PK most strong peaks in the wavetrain processed.

11.6 Program "estimtt": estimating travel times for regional and teleseismic events

Example of input-file

```
****PROGRAM ESTIMTT: ESTIMATING TRAVEL TIMES FOR REGIONAL AND
TELESEISMIC EVENTS****
ID OF ARRAY  (NRS,ARC,FIN,APA,GRS)
alibek
COORDINATES OF ARRAY  (Latitude, Longitude - grad.)
    37.92933    58.1125
FLAG OF USED TRAVEL TIME TABLE:
( '0' - Only for distance <30 grad. - NORESS regional travel time table
)
( '1' - For distance from 0 to 180 grad. - Seismological travel time
table by JEFFRIES and BULLEN.
 If distance >30 grad. - only '1' by default)
1
NUMBER OF PHASES
12
ID OF PHASES:
(PN,PG,SN,SG,LG,RG - if NORESS regional travel time table is using)
(P,PP,PPP,PCP,PKP1,PKP2,S,SS,SSS,SCS,LGT,LR - if travel time table by
JEFFRIES and BULLEN is using)
PN
PG
SN
SG
LG
RG
LATITUDE OF EPICENTER (grad.)
41.5000
LONGITUDE OF EPICENTER (grad.)
 88.7000
DEPTH OF EVENT
```

```
0.
ORIGIN TIME (in form at YYYY-DDD:HH.MM.SS.mmm)
1994-123:03.04.34.00
FLAG OF USED SCRIPT (0-use SA, 1-use script)
0
```

DESCRIPTION OF PROGRAM "ESTIMTT"

The program computes values of the travel-times and azimuths for desired regional or teleseismic phases arrived on array from the seismic event. The information used for the calculations is: event location, its origin time and travel-time curves for shallow source (depth isn't used in this calculations). For each valid phase type, this program requires a corresponding travel time table. The regional travel-time tables are in the file "trreg.tm". Teleseismic travel-time tables are in the file "trtel.tm".

This procedure:

1) reads parameters from input-file 'estimtt.inp';

2) converts character string YYYY-DDD:HH.MM.SS.mmm of origin time to CSS epoch time. Epoch time is double precession floating point seconds from firstday of 1970. (I.e. epoch time 0.000 is 00:00:00 1970);

3) computes distance and azimuth to the station (forward problem for azimuth); calculate angular distance, azimuth and back azimuth between two point on a Crassovski Earth's ellipsoid; azimuth and back azimuth are positive and measured clockwise from local north;

4) reads files with the travel time tables (for each valid regional and teleseismic phase types, this program requires their travel time tables contained in the files "trreg.tm" and "trtel.tm" correspondingly); finds travel time for given distance and the associated phase; computes arrival times for phases in CSS epoch time;

5) converts values of arrival times from the CSS epoch time to printable character string, i.e. human time YYYY-DDD:HH.MM.SS.mmm.

Subroutines called : obg, rdsimar, origtm, starsub

DESCRIPTION OF INPUT FILE PARAMETERS

ID OF ARRAY

The array identificator; possible value is one of the strings: 'NRS', 'ARC', 'FIN', 'APA', 'GRS'

COORDINATES OF ARRAY
The latitude (in degrees from -180 to +180) and Longitude (in degrees from -90 to +90) of array

FLAG OF USED TRAVEL TIME TABLE:
'0' - (fixed for distance <30 grad.) means that the NORESS regional travel time table is used. Phase identificators currently valid for the regional phases are: 'PN', 'PG', 'SN', 'SG', 'LG' or 'RG'.
'1' - (fixed for distance from 0 to 180 grad.) means that the Seismological travel time table by Jeffries and Bullen is used. Phase identificators currently valid for the teleseismic phases are: 'P', 'PP',' PPP', 'PCP', 'PKP1', 'PKP2', 'S', 'SS', 'SSS', 'SCS', 'LGT', 'LR'. If distance >30 grad., then only '1' is set by default.

NAMBER OF PHASES
The total number of phases for this event. Values less or equal 6 are permitted - for regional phases, and less or equal 12 - for teleseismic phases.

ID OF PHASES
The identificators for selected phases (arranged as the column).

LATITUDE OF EPICENTER
The latitude of epicenter (in degrees from -90 to +90)

LONGITUDE OF EPICENTER
The longitude of epicenter(in degrees from -180 to +180)

DEPTH OF EVENT
The focal depth of epicenter (in km) (Now this parameter is assumed = 0.)

ORIGIN TIME
The origin time is a character string in the standard time format, i.e.
1989-018:09.01.55.123:
Positions 1 - 4 are the Year; position 5 is the symbol '-'; positions 6 - 8 are the Day-of-year; position 9 is the symbol ':'; positions 10 - 11 are the Hours; position 12 is the dot symbol '.'; positions 13 - 14 are the Minutes; position 15 is the dot symbol '.'; positions 16 - 17 are the Seconds; position 18 is the dot symbol '.' position 19 - 21 are the Milliseconds.

FLAG OF USED SCRIPT
'0' means that program is used in the interactive SNDA mode; '1' means the use of the program in a script mode.

OUTPUT PARAMETERS OF THE PROGRAM:

Arrival times of the phases. Each of these outputs is the character string in the standard time format, i.e. 1989-018:09.01.55.123

Travel times of phases (in sec).

Azimuth from station to epicenter.

Distance from station to epicenter.

11.7 Program "arloc": event locating based on single array data

Example of input-file

```
****PROGRAM ARLOC: EVENT LOCATING USING SINGLE ARRAY DATA*****
ID OF ARRAY (NRS,ARC,FIN,APA,GRS)
NRS
COORDINATES OF ARRAY (Latitude, Longitude)
 60.735000      11.541000
NUMBER OF PHASES
 6
ID OF PHASES  (PN,PG,SN,SG,LG,RG)
PN
PG
SN
SG
LG
PG
ARRIVEL TIMES OF PHASES
1994-123:03.09.12.619
1994-123:03.10.34.388
1994-123:03.12.37.016
1994-123:03.15.35.088
1994-123:03.15.10.442
1994-123:03.10.29.697
AZIMUTHS OF PHASES
    180.769200
    180.703200
    182.111100
    168.445300
    174.462400
    189.066400
APPARENT VELOCITIES OF PHASES
   7800.413000
```

```
   6198.923000
   4599.069000
   3599.630000
   3550.551000
   6199.543000
STANDART DEVIATION OF TRAVEL TIME
  2.500000E-01     1.300000     2.500000     3.500000
    4.200000     5.300000
STANDART DEVIATION OF AZIMUTH
    2.000000     4.000000     6.000000     8.000000
    9.000000    10.000000
STANDART DEVIATION OF  VELOCITY
    5.000000E-01 7.000000E-01 9.000000E-01 9.000000E-01
    1.000000     1.000000
```

DESCRIPTION OF PROGRAM "ARLOC"

The program: locates events using information about azimuths and arrival times for seismic phases associated from array data. The event epicentre coordinates and confidence bounds for them are evaluated only for regional distances. The epicentre distance is calculated using all possible combinations of arrival times for all pairs of seismic phases involved.

The program performs the following :

1) reads the input file 'arloc.inp';

2) converts character strings YYYY-DDD:HH.MM.SS.mmm of arrival times to epoch times; the epoch time is double precision floating point seconds from the first day of 1970;

3) estimates a set of event epicentre distances for all the pairs of associated wave phases using the differences of phase arrival times and the corresponding travel time tables. Then it calculates the mean value and weighted (by variances of phase travel times errors) mean value for the epicentre distance; the travel-time tables are stored in the file 'deltr.tm';

4) estimates the mean and weighted (by error variances) mean values fot the event azimuth;

5) calculates the coordinates of epicenter: the latitude and longitude in degrees, using the mean and weighted mean values of distance and azimuth. (the angular distance, azimuth and backazimuth between two points on a Crassovski Earth's ellipsoid are really calculated. The azimuth and back azimuth are assumed to be always positive and measured clockwise from a local North.);

6) evaluates the travel times of phases involved in accordance with the estimated epicentre distance. For each valid regional phase, the program uses

the specific travel time table; the travel-time tables are contained in the file 'trreg.tm';

7) computes the approximate origin epoch time in the CSS format and converts it to the printable character string, i.e. the "human time" format YYYY-DDD:HH:MM:SS.mmm.

The Pn and Sn travel-time tables used in the program were computed for NORESS array with the velocity model by S. Mykkeltveit. For this model the next parameters of the medium and regional wave velocities are used:

Thickness(km)	Vp (km/sec)	Vs (km/sec)
16,0	6,20	3,58
24,0	6,70	3,87
150,0	8,19	4,60
> 150,0	8,23	4,68

The group velocities for the next phases are assumed to be constant: Pn - 6,20 km/sec, Lg - 3,55 km/sec, Rg - 3,00 km/sec.

The program calls the following subroutines : prg, obg, dinp, origtm, distdtr, wrtarlc, starsub

DESCRIPTION OF INPUT FILE PARAMETERS

ID OF ARRAY
The identificator of array; a possible value is one of the string: 'NRS', 'ARC', 'FIN', 'APA', 'GRS'
COORDINATES OF ARRAY
The latitude (in degrees from -180 to +180) and longitude (in degrees from -90 to +90) of the array

NUMBER OF PHASE
The total amount of associated. phases for the event. A value less or equal 6 is permitted (for regional phases only!).

ID OF PHASES
The character string identificator for each phase involved. The valid ID strings are: 'PN', 'PG', 'SN', 'SG', 'LG', 'RG'.

ARRIVAL TIMES OF PHASES

The arrival times for each phase involved . The arrival times have to be given as character strings in the standard time format, i.e. 1989-018:09.01.55.123: Positions 1 - 4 are the Year; position 5 is the symbol '-'; positions 6 - 8 are the Day-of-year; position 9 is the symbol ':'; positions 10 - 11 are the Hours; position 12 is the dot symbol '.'; positions 13 - 14 are the Minutes; position 15 is the dot symbol '.'; positions 16 - 17 are the Seconds; position 18 is the dot symbol '.' position 19 - 21 are the Milliseconds.

AZIMUTHS OF PHASES
The azimuths from array to the source for every phase involved, arranged as the column (in degrees from 0 to +360).

APPARENT VELOCITIES OF PHASES
The apparent velocities for every phase involved, arranged as the column (in km/sec).

STANDARD DEVIATIONS OF TRAVEL TIME
The expected (a'priory) mean square errors of onset times for every phase involved, arranged as the column (in sec.).

STANDARD DEVIATIONS OF AZIMUTH
The expected (a'priori) mean square errors of azimuth estimates for every phase involved, arranged as the column (in degrees).

STANDARD DEVIATIONS OF VELOCITY
The expected (a priori) mean square errors of apparent velocity estimates for every phase involved, arranged as the column (in km/sec).

OUTPUT PARAMETERS OF THE PROGRAM:

Latitude of epicenter (in degrees from -90 to +90)
Longitude of epicenter (in degrees from -180 to +180)
Origin time: character string in standard time format,
i.e. 1989-018:09.01.55.123
Azimuth from epicenter to station.
Distance from epicenter to station.
Length of Large semi-axis of confidence ellipse on 0.9 error level.
Length of Small semi-axis of confidence ellipse on 0.9 error level.

11.8 Program "ldst": learning data statistics evaluation

Example of input-file

```
**** PROGRAM LDSTST: LEARNIG DATA STATISTICS*****
NAME OF FILE FOR GLOBAL INFORMATION
data/glob.dat
NUMBERS OF STACK CHANNELS
all
NAME OF FILE FOR COVARIANCE MATRIX AND MEAN VECTORS OF FEATURES
./data/resku.dat
NUMBER OF THE BASIC FEATURE FOR SCATTERPLOTS
10
NAME OF FILE FOR GRAPHIC RESULTS
plot/gr.gr
```

DESCRIPTION OF PROGRAM "LDSTST" (Learning data statistic)

The program "ldstst" accomplishes the following procedures:

1) reads learning vectors from the System SNDA Stack;

calculates: a) the mean vectors m(k) for each class k = 1,2,...,M , b) the (p x p) matrix R with elements r(jl) which are the estimates of correlation coefficient between features with numbers j and l, c) the sample covariance matrix C;

2) types on the screen: a) the mean vectors m(k), b) the correlation matrix R;

3) performs the plotting scattering diagrams of the features with the help of standard UNIX routine "plotxy".

The mean vectors m(k) are calculated with the formula

$$m(k)= (1/nk)(Xk(1) + ...+ Xk(i) + ...+ Xk(nk)),$$

where : Xk(i) -is the i-th feature vector from class k ; i = 1,2,...,nk ; k = 1,2,...,M .

The elements r(jl) of matrix R are defined by the formula

$$r(jl) = c(jl) / sqrt(c(jj)\ c(ll)) ,\ j,l = 1,2,...,p.$$

where

c(jl) - elements of unnormaliezed covariance matrix C:

$$C = SUM(k=1,M)\{SUM(i=1,nk)[(Xk(i)-m(k))* (Xk(i)-m(k))]\}$$

* is the sign of vector transposition.

In geometrical sense m(k) are the "centres" of classes in the p-dimensional feature space. The large distance between the classes implies the low error probabilities while discriminating. The analysis of matrix R allows one to reveal groups of uncorrelated features and/or pairs of strong correlated features. If an absolute value of a coefficient r(jl)>0.9 , then one of the features: j or l, may be eliminated.

Scatterplots for all the pairs of features can be exposed on the screen with the help of standard UNIX routine "plotxy". This plotting is managed in the following manner: one of the features is being chosen as the "basic", and p-1 plots with scattering diagrams of this feature with the remaining ones are displayed on the screen. At each plot the points are drown on the screen for given features pair: the values of the "basic" feature are set along the X-axis for all the observations; the values of the another feature for given pair - along the Y-axes. The points corresponding to different classes are plotted by different symbols. As a result, some clusters of points attributed to different classes can be seen on the diagrams. From analysis of the scatterplots one may select the most separated pairs of features. Besides that, it is possible to estimate the similarity of two-dimensional distributions of features to the normal distribution .

DESCRIPTION OF INPUT FILE PARAMETERS

NAME OF FILE FOR GLOBAL INFORMATION (input file)
The structure of this file is described in the program LD

NUMBERS OF STACK CHANNELS (input information)
This row corresponds to channels chosen for the analysis . If the all channels are kept, "all" must be written in this row. In other cases numbers of kept channels are given according to rules of the Stack commands.

NAME OF FILE FOR COVARIANCE MATRIX AND MEAN VECTORS OF FEATURES (output file)
This file contains: the mean vectors m(j), j=1,...,M for each class; normalised correlation matrix R ; covariance matrix C; some service information needed for execution of the following programs

NUMBER OF THE BASIC FEATURE FOR SCATTERPLOTS

The feature is used for design of p-1 scattering diagrams. On the every diagram this feature values are set along the X axis, and along the Y-axis - the values of one from remaining features

NAME OF FILE FOR GRAPHIC RESULTS
This is the name of command file for the "plotxy" UNIX routine.

11.9 Program "fsel": selection of informative features

Example of input-file

*********PROGRAM FSEL: SELECTION OF INFORMATIVE FATURES********
NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS
/data/resku.dat
NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES
/data/select.dat
NAME OF FILE FOR GRAPHIC RESULTS
plot/mygr.gr

DESCRIPTION OF PROGRAM "FSEL" (Features selection)

The program "fsel" accomplishes the following operations:

1) realises an automatic stepwise selection of the most informative features providing the least error probabilities for the classification based on given set of learning data;

2) constructs a function R(k), k = 1,2,...,p, which is a cumulative over all pairs of classes Mahalanobious distance in depend on a selection step number k;

3) for the case of two classes constructs a function P(k), k=1,...,p which is a theoretic probability of total classification errors in depend on an amount k of features used for classification;

4) calculates value ko for which the function P(k) attains its minimum (ko=argmin P(k));

5) plots the calculated functions on the screen using the standard UNIX routine "plotxy" and types on this plot the numbers and labels of features chosen at the each selection step.

The Mahalanobious distance between two p-dimensional probability distributions (with numbers l and s) is defined by the quadratic form :

$$MD(k,l,s) = (m(k,l) - m(k,s))^* \, Sinv(k) \, (m(k,l) - m(k.s))$$

where:
m(k,l), m(k,s) - are the sample mean vectors of classes l and s for some k selected features; S(k) - is the sample covariance matrix calculated for this features using learning data for both the classes; Sinv(k) means the inverse matrix for S(k).

The cumulative (over all the pairs of classes) Mahalanobious distance is defined by the formula

$$R(k) = \mathrm{Sum}(l=1,M)\{\mathrm{Sum}(s=1,M)[MD(k,l,s)]\},$$

where M is the number of classes.

At the first step of the selecting procedure p values of the R(1) functional are calculated for every feature. The maximum from these p values is attained at some j(1) feature which is thus selected. At the second step p-1 values of the R(2) functional are calculated for the pairs of features: the first member in this pairs is always the previously selected feature j(1), the second member - is an arbitrary feature from the rest ones. Then the second feature is selected which ensure the maximum of these R(2) values. At the k-th step of this selecting procedure p-k+1 values of the R(k) functional are calculated for a set of feature vectors. The first k-1 components in these vectors are the features which were selected at the previous steps, the k-th component is an arbitrary feature from the remaining ones. On the each step k (k = 1,2,...,p) of the selecting procedure the number and label of selected feature are typed on the screen and the theoretical value of total error probability P(k) is calculated. The value P(k) (for k>1) is calculated by the formula:

$$P(k) = (1/2)[1 - Tk(R(k)/sigma(k)) + Tk(-R(k)/sigma(k))],$$

where

$$[sigma(k)]^{**}2=[(t+1)/t][r1+r2+R(k)];\ t=[(r1+r2)/r1\ r2]-1;\ r1=k/n1;\ r2=k/n2$$

$$Tk(z) = F(z) + (1/(k-1)) * (a1 - a2*H1(z) + a3*H2(z) - a4*H3(z)) f(z),$$

F(z) - is the distribution function of standard normal distribution , f(z) - is the probability density function of this distribution; Hi(z) is the Hermitian polynomial with order i, i=1,2,3; aj, j=1,...,4 are some coefficients depending on k, n1, n2 and R(k).

This formula was derived via an asymptotic expansion of the distribution function for commonly used linear discriminator. The assumption

was used that the number of features p and numbers of learning vectors n1, n2 for both the classes are simultaneously increasing with the same rate.

The program "fsel" then defines a number ko of that selecting step for which a minimum of function P(k), k=1,...,p, is attained: ko = argmin P(k). Thus the optimal set of features with the numbers j(1), j(2),..., j(ko) become determined. These features provide the minimal total error probability.

DESCRIPTION OF INPUT FILE PARAMETERS

NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS (input file):
The file contains the covariance matrix and mean values of the features, calculated by the program "ldst".

NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES (output file):
The file contains : the amount of optimal features selected; the mean vectors of selected features in the order of their selection by the program; the covariance matrix of selected features.

NAME OF FILE FOR GRAPHIC RESULTS (output file):
This file is the command file for the “plotxy” UNIX routine. The graphs of functions ro(k), P(k) , along with the numbers and labels of chosen features are exposed on the screen .

STANDARD OUTPUT FILES FOR PROGRAM RESULTS

File "perr.dat": The file contains values of classification error probabilities, corresponding to selected features in the order of their selection in the program.

File "jnum.dat":The file contains the selected feature numbers in the order of their selection in the program.

File "sumr.dat":The file contains values of Mahalanobious distances corresponding to selected features in the order of their selection in the program.

11.10 Program "ldiscr": linear discriminator for statistical classification

Example of input-file

******PROGRAM LDISCR: LINEAR DISCRIMINATOR FOR CLASSIFICATION ******
MODE FOR FEATURE SELECTING: 0 - WITHOUT SELECTING, 1 - WITH SELECTING
0
NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS
./data/resku.dat
NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES
/data/select.dat
NAME OF FILE FOR VECTORS TO BE CLASSIFIED
data/newvect.dat
NAME OF FILE FOR RESULT OF CLASSIFICATION
data/ldres.dat

DESCRIPTION OF PROGRAM "LDISCR" (Linear discriminator)

The program "ldiscr" classifies given feature vectors X1 ,...,Xr as belonging to one of the M classes. The classification is accomplished with the Linear Discrimination Function. The input data for the program consists of: mean vectors m(k) for every class k=1,2,...,M, a feature covariance matrix S, the same for both the classes, and input feature vectors X 1,...,Xr to be classified. For each class k=1,2,...,M, the program calculates "informants":

$$T(k) = X*Sinv(m(k) - (1/2)\, m(k)*\, Sinv\, m(k) \quad k = 1,2,...,M.$$

where: * is the sign of vector transposition; Sinv - is inverse matrix for S. Then the program calculates number ko providing a maximum for the values T(k) k=1,...,M and thus classifies the vector X as belonging to the class with number (ko). For the case M = 2 , the classification procedure can be modified. For this case the statistics

$$Q(k) = [R/(R+1)]T(k) + rk/2, \quad k=1,2$$

are calculated, where R = (ro/(r1)(r2)) - 1; ro = r1 + r2; r1 = p/n1; r2 = p/n2 p - is the total number of features, n1 - is the number of observations of 1-st class, n2 is the number of observations of 2-nd class. If Q(2) > Q(1), then the program classifies the vector X as belonging to class 2, othewise - to class 1.

The program "ldiscr" has two options : "0" and "1", for classification using either all the features or only the features selected by the program "fsel".

DESCRIPTION OF INPUT FILE PARAMETERS

MODE FOR FEATURE SELECTING: 0 - WITHOUT SELECTING, 1 - WITH SELECTING [MODSWT] (input parameter)
If MODSWT=0, the program uses for classification the initial set of all features, if MODSWT=1 , only the features selected by the program "fsel" are used by the classification statistic.

NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS (input file)
This file is used for reading parameters of the linear discrimination function in the case MODSWT=0. The file is the output of the program "ldstst".

NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES (input file)
This file is used for reading parameters of the linear discrimination function in the case MODSWT=1. The file is the output of the program "fsel".

NAME OF FILE FOR VECTORS TO BE CLASSIFIED (input file)
This file contains the "new" vectors with numbers 1,...,no, which have to be classified by the program.

NAME OF FILE FOR RESULTS OF CLASSIFICATION (output file)
The file contains the table with numbers of "new" vectors and numbers of classes corresponding to this vectors in the result of classification.

11.11 Program "exam": estimation of probability of errors by moving classification of learning data

Example of input-file

******PROGRAM EXAM: CLASSIFICATION OF LEARNIND DATA******
MODE FOR FEATURE SELECTING: 0 - WITHOUT SELECTING, 1 - WITH SELECTING
0
NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS
data/resku.dat
NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES
/data/select.dat
NAME OF FILE FOR GRAPHIC RESULTS
plot/exam.gr

DESCRIPTION OF PROGRAM "EXAM" (Examination)

The program "exam" calculates by the known Jack-Knife method the amounts of misclassification for every pair of classes. On this basis the estimates of classification error probabilities are evaluated. The values of linear discrimination function (LDF) corresponding the learning vectors are ranked in their magnitude for both the classes The ranked LDF are displayed on the screen.

The estimate nu(i,j) of classification error probability are obtained in the program with the formula

$$nu(i,j) = n(i.j) / n(j) ,$$

where: n(i.j) - is an amount of vectors from class j attributed by the LDF to class i, n(j) - is a total amount of vectors from class j .

The averaged estimate of classification error probability is defined by the formula

$$nu = (1/M)SUM(j=1,M)\{SUM(i=1,M;\ i=/=j)[nu(i.j)]\}.$$

The estimates nu(i.j) and nu are calculated by the known Jack-Knife method ("moving examination" method). In this method on each step one of the learning vectors is eliminated from the learning set. The remaining vectors are used as the learning data for calculating the LDF. The eliminated vector is then classified by this LDF. If this vector is classified incorrectly, the value

n(i.j) is increased by the unit. The eliminated vector is then returned into the learning set and the next vector is extracted. This procedure is repeated with all the learning vectors. Values nu(i,j) and nu evaluated by the Jack-Knife method are asymptotically unbiased estimates of the classification error probabilities.

The program "exam" calculates sequences L1(i), i=1,...,n1 and L2(j),j=1,...,n2 of LDF values for the observations from the two classes. These sequences are ranked in the order of their magnitudes and plotted on the screen. Values L1(i) are marked by the symbol "+", values L2(i) are marked by the symbol "o". The plots of L1(i) and L2(j) functions enable one:

to detect incorrectly classified observations;

to select the observations corresponding to the "uncertain" classification area defined by the condition :-0,5 < L(i) < 0,5 .

The amount of observations belonging to the "uncertain" classification area is an important characteristic of classification performance.

DESCRIPTION OF INPUT FILE PARAMETERS

MODE FOR FEATURE SELECTING: 0 - WITHOUT SELECTING, 1 - WITH SELECTING [MODSWT]

If MODSWT=0, the program uses the initial set of all features for the classification, if MODSWT=1 , only the features selected by the program "fsel" are used by the classification statistic.

NAME OF FILE FOR COVARIANCE MATRIX AND MEAN FEATURE VECTORS

This file is used for reading parameters of the linear discrimination function in the case MODSWT=0. The file is the output of program "ldstst".

NAME OF FILE FOR COV. MATRIX AND MEAN VECT. FOR SELECTED FEATURES

This file is used for reading parameters of the linear discrimination function in the case MODSWT=1. The file is the output of program "fsel".

NAME OF FILE FOR GRAPHIC RESULTS

This is the command file for the "plotxy" UNIX routine. The Linear Discrimination functions in depend on numbers of learning observation vectors are plotted for both classes by this routine.

References

1. Aki, K., Richards, P.,1980, Quantative seismology. Theory and Methods, Freeman and Comp., San-Francisco.

2. Baer,M. and Kradofler,U., 1987, An automatic phase picker for local and teleseismic events: Bull. Seism. Soc. Am., vol. 77, p.1437-1445.

3. Bache,T.C., Bratt,S.R., Wang,J, Fung,R.M., Korbin,C., Given,J.W.,1990, The intelligent monitoring system: Bull. Seism. Soc. Am., vol. 80, p.1833-1851.

4. Bache,T.C. and SAIC staff, 1990, Inelligent Array System. System introduction and functional description: Report of Sciense Application International Corporation, San Diego, California.

5. Backus,M., Burg,J., Boldwin,D and Bryan,E., 1964, Wide-band extraction of mantle P-waves from ambient noise, Geophysics, v.29, p.672-692.

6. Bame,D.A., Walck M.C., and Heibert-Dodd K.L., 1990, Azimuth estimation capabilities of the NORESS regional seismic array: Bul. Seism. Soc. Am., vol 80, p.1999-2016.

7. Benioff,H.,1935, A linear strain seismograph, Bul. Seism. Soc. Am. V2, N4.

8. Bannister, E.S., E.S.Husebye and B.O.Ruud, 1990, Teleseismic P coda analysis by three-component and array techniques: deterministic location of topographic P-to-Rg scattering near noress array, Bull. Seism. Soc. Am., 80,
1969-1986.

9. Bessonov, A.F., Iakovlev, A.P., Komotskiy,V.A., Nikulin,V.F.,1992, Broad-band strainmeter with opto-electronic displacement transducer. Proceedings 7th Inter. Symposium Geodynamics and Physics, IAG-Symposium, Potsdam, Germany.

10. Box,G., and Jenkins,G, 1970, Time series analysis. Forecasting and control: Holden-day, Amsterdam, 406 p..

11. Bratt,S.R, and Bach,T.C., 1988, Locating events with a sparse network of regionzl arrays: Bull. Seism. Soc. Am. 78, p.780-798.

12. Bratt, S.R., H.J. Swanger, R.J.Stead, F. Ryall, and T.C. Bache, 1990, Initial results from the intelligent monitoring system, Bull. Seism. Soc. Am. , 80, 1852-1873.

13. Brilliinjer, D., 1976, Time series. Theory and data processing. J. Willy and sons, N-Y.

14. Bungum,H., Husebye,E., and Ringdal,F., 1971, The NORSAR array and preliminary results of data analysis: Geophis. J. Roy. Astron. Soc., N 25, p.115-126.

15. Burg, J.P., 1964, Three-dimensional filtering on array of seismometers, Geophysics, v.29, p.693-713.

16. Cassidy,F., Cristofferson,A, Husebye,e.S., and Ruud,B.O., 1990, Robust and reliable techniques for epicenter location using time and slowness observations: Bul. Seism. Soc. Am., vol. 80, p.140-149.

17. Capon,J., Greenfild,R., Kolker,R. and Lacoss,R., 1968, Short-period signal processing results for large aperture seismic array: Geophysics, vol. 33, p.452-472..

18. Capon,J., 1969, High resolution frequency-wavenumber spectral analysis: Proc. IEEE, vol. 57, p.1408-1418.

19. Capon,J., 1970, Application of space-time domain decision and estimation theory to antenna processing system design: Proc. IEEE, vol. 58, p.170-180.

20. Claassen,J.P., 1992, The application of multipy constrained minimum variance adaptive beamforming to regional monitoring. Bul. Seism. Soc. Am., vol. 82, pp. 2191-2212.

21. Cristoffersson,A., Husebye,E.S., and Ingate,S.F., 1988, Wavelet decomposition using ML probabilities in modeling single site 3-component records: Geophys. J. Int, p.197-213.

22. Der,Z.A., Hirno,M.R. and Shumway,R.H., 1990, Coherent processing of regional signals at small seismic arrays: Bul. Seism. Soc. Am., vol 80, p.2161-2177.

23. Gupta, I.N., C.S. Lynnes and R.A. Wagner, 1990, Broadband F-K analysis of array data to identify sources of local scattering, Geophys. Res. Letters, 17, 183-186.

24. Hannan, E.J., 1970, Multiple time series: J.Wiley & Sons, New York, 575p.

25. Hansen,R.A., Ringdal,F., and Richards,P.G.,1990, The stability of RMS L-measurements and their potential for accurate estimation of the yields of Soviet underground nuclear explosions: Bull. Seism. Soc. Am., vol 80, p.2106-2126.

26. Harris D.B., 1990, Comparison of the direction estimation performance of high-frequency seismic arrays and three-component stations: Bul. Seism. Soc. Am., vol 80, p.1951-1968.

27. Haykin, S. (editor), J.H. Justic., N.E. Owsley, J.L. Yen, A.C. Kak, 1985, Array Signal Processing, Prentice-Hall, Inc., N.Y.

28. Hjortenberg, E., Risbo, T.,1975, Monochromatic components of seismic noise in NORSAR area, Geoph. J. Roy. Astron. Soc., v.42, p.547-554.

29. Husebye, E.S. and B.O. Ruud, 1989, Array seismology: past, present and future developments, in J.L. Litchiser (ed.), Observational Seismology, Univ. California Press, Berkley, Ca., USA, 123-153.

30. Ingate,S.F., Husebey,E.S., and Christofferson,A., 1985, Regional array and optimal processing schemes: Bul. Seism. Soc. Am., vol. 75, p.1155-1177.

31. Iakovlev, A.P., Alyoshin,V.A., 1994, Study of monochromatic spectral lines in high-frequency seismic noise, Fizika Zemli, N3, p.3-19.

32. Jepsen D.C. and Kennet B.L.N., 1990 Three-component analysis of regional seismograms: Bul. Seism. Soc. Am., vol 80, p.2032-2053.

33. Jurkevis,A., 1988, Polarisation analysis of the three-component array data: Bul. Seism. Soc. Am., vol. 78, p.1725-1743.

34. Kennet B.L.N., 1980, Seismic wave propagation in stratified media, Cambridge Univercity Press, 340 pp,

35. Kushnir, A.F., V.F. Pisarenko and T.A. Rukavishnikova, 1980, Noise compensation in multidimensional geophysical observations. 1. Theory and methods of data processing, (Computational Seismology, 13), Allerton
Press, Inc., 146-151.

36. Kushnir, A.F., I.V. Nikiforov and I.V. Savin, 1983, Statistical adaptive algorithms for automatic detection of seismic signals, (Computational Seismology, 15), Allerton Press, Inc., 145-162.

37. Kushnir, A.F. and V.M. Lapshin, 1984, Optimal processing the signals received by the group of spatially distributed sensors, (Computational seismology, 17), Allerton Press, Inc., 163-174.

38. Kushnir, A.F. and Lapshin,V.M. , 1986, Parametric methods of analysis of multiple time series, M, Nauka, Publ. Inst. Phys. Earht, Acad. Sci. USSR, 242 pp.,148 references (in Russian).

39. Kushnir, A. F.,1989, Asymptotically optimal statistical analysis of geophysical field, Dr. Sc. Dissertation. Moscow, IPE Ac.Sc. USSR, 380 pp., 218 references (In Russian).

40. Kushnir, A.F., Mostovoy, C.V.,1990, Statistical processing of geophysical fields, Naukova Dumka, Kiev, 293pp., 124 references (in Russian).

41. Kushnir, A.F., Lapshin, V.M., Pinsky, V.I. and Fyen, J, 1990, Statistically optimal event detection using small array data: Bull. Seism. Soc. Am., vol. 80, p.1934-1947.

42. Kushnir, A.F., Pinsky, V.I., Tsvang, S.L., Fyen, J., Makkeltveit, S. and Ringdal, F. , 1990, Optimal group filtering and noise attenuation for NORESS and ARCESS arrays: Semianual Technical summary, NORSAR Sci. Rep. 1-90/91, Kjeller, November 1990, p.115- 134.

43. Kushnir, A.F., Fyen, J. and Kvarna, T., 1991, Studing of multichannel statistical data processing algorithms in the framework of the NORSAR event processing program package: Semiannual Technical Summary, NORSAR Sci. Rep. 2-90/91, Kjeller, p.82-103.

44. Kushnir, A.F. and Gulko, E.A, 1991., Statistical optimization of seismic holography algorithms for array data processing: Semiannual Technical Summary, NORSAR Sci. Rep., no 1-90/91, Kjeller, Norway, November 1991, p.62- 73.

45. Kushnir, A.F., and Kvarna, T., 1991, Initial testing of mixed event separation using statistically optimal adaptive algorithm, Semiannual Technical Summary, NORSAR Sci. Rep. no 1-91/92, Kjeller, Norway, p 112 - 126.

46. Kvarna, T. and Ringdal, F., 1992, Integrated array and three-component processing using a seismic microarray: Bul. Seism. Soc. Am., vol. 82, p.

47. Kvarna,T. 1989, On explotation of small apeture NORESS type arrays for enhanced P-wave detectability: Bul. Seism. Soc. Am. v.79, p.888-900.

48. Kvarna,T., Kibsgaard,S. and Ringdal,F., 1987, False alarm studies and treshold determination for regional event detection: Semianual Techn. Summary, 1 Apr.- 30 Sept.,1987, NORSAR Sci. Rept. N1-87/88 Kjeller, Norway.

49. Kvaerna, T., 1993, Intelligent post-processing of seismic events. Part 2: Accurate determination of phase arrival times using autoregressive likelihood estimation, NORSAR Sci. Rep., no 2-92/93, Kjeller, 68-92.

50. Kvaerna, T., 1993, A generic algorithm for accurate determination of P-phase arrival times, Semiannual Technical Summary, NORSAR Sci. Rep., no 2-93/94, Kjeller, Norway, November 1993, 98-108.

51. Lehmann, E.L., 1960, Testing statistical hypotheses, John Willey, Inc., New York.

52. Lienert,B.R., Berg.E, and Frazer,L.N., 1986, Hypocenter: an earthquake location method using centered, scaled and adaptively damped least squares, Bull. Seism. Soc. Am., vol. 76, p.771-783.45.

53. Lokshtanov,D.E., Ruud,B.O., and Husebye,E.S., 1991, The crustal transfer function in seismic three-component slowness estimation, Geophys. Res. Lett.. 18, p.1393-1396.

54. Mykkeltveit,S.and Bungum,H., 1984, Processing of regional events using data from small-aperture arrays: Bull. Seism. Soc. Am., vol. 74, p.2313-2333.

55. Mykkeltveit,S., Ringdal,F., Kvarna,T. and Alewine,R., 1990, Application of regional arrays in seismic verification research: Bull. Seism. Soc. Am., vol. 80, p.1777-1800.

56. Pinsky,V.I., Tsvang,S.L. and Fyen,J., 1991, On-line detection using adaptive statistically optimal algorithms: Semiannual Technical Summary, NORSAR Sci. Rep. 1-91/92, Kjeller, November 1991, p.82- 99.

57. Pisarenko,V.F., Kushnir,A.F. and Savin,I.V., 1987, Statistical adaptive algorithms for estimations of onset moments of seismic phases: Phys. Earth Planet. Interiors, vol. 47, p.888-900.52. Ringdal,F. and Husebye.E.S., 1982, Application of arrays in the detection, location and identification of seismic events: Bul. Seism. Soc. Am., vol. 72, p.2201-2224.

58. Peterson, J.,1993, Observation and modelling of seismic background noise, Open-file report, 1993, Albuquerque, NM.

59. Ringdal,F.,1990, Teleseismic event detection using the NORESS array, with special reference to low-yield Semipalatinsk explosions: Bul. Seism. Soc. Am., vol 80, p.2127-2143.

60. Ringdal, F. and E.S. Husebye, 1982 Application of arrays in the detection, location and identification of seismic events, Bul. Seism. Soc. Am., 72, 2201-2224.

61. Ringdal,F. and Kvarna,T., 1989 A multi-channel processing approach to real time network detection, phase association, and threshold monitoring: Bull. Seism. Soc. Am., vol. 79, p.1927-1940.

62. Roberts,R.G., Christofferson,A. and Cassidy,F., 1989, Real time event detections, phase identificatons and source location stimation using single station three-component seismic data: Geophys. J. Int., 97, p.471-480.

63. Roberts,R.G.,and Cristofferson,A., 1990, Decomposition of complex single-station three-component seismograms: Geophys. J. Int., 103, p.57-74.

64. Ruud,B.O., 1990, Teleseismic epicenter locations from arrival times a regional networks: Jeophys. Jorn. Int., 100, p.515-519.

65. Ruud,B.O., Husebye,A., Christofferson.A., and Ingate,S.F., 1988, Event location at any distance using seismic data from single, three-component station: Bul. Seism. Soc. Am., vol. 78, p.308-335.

66. Ruud,B.O., and Husebye,E.S., 1992, A new three-component detector and automatic single-station bulletin production: Bul. Seism. Soc. Am., vol. 82, p.21-237.

67. Sambridge,M.S. and Kennet,B.L.N., 1986, A novel method of hypocentre location: Grophys. J.R.Astr.Soc., 87, p.679-697.

68. Shen,W.W., 1979, Aconstrained minimum power adaptive beamformer with time varying adaptation rate, Geophysics, v.44, p.1088-1096.

69. Shopland, R.C., Kirklin, R.H.,1970, Application of a vertical strain seismograph to the enhancement of P waves, Bul, Seism. Soc. Am., v.60, n1.

70. Sorrels, G.G.,1971, A preliminary investigation into the relationship between long-period seismic noise and local fluctuations in the atmospheric pressure field, Geophys. J.,v.26,p.71-82.

71. Suteau-Henson A., 1990, Estimation azimyuth and slownes from three-component and arrays stations: Bul. Seism. Soc. Am., vol 80, p.1987-1998.

72. Tarvainen,M., 1992, Automatic seismogram analysis: statistical phase picking and locating method using one-station three-component data, Bull. Seism. Soc. Am., 82, p.860-869.

74. Thuberg,C.H., Given,H. and Berger,J. 1989, Regional seismic event location with a sprase network: Application to eastern Kazakhstan: J. Geophys. Res., 94, p.17767-17780.

75. Troitsky,P.N., Husebye,E.S. and Nikolaev,A.V., 1981, Lithospheric studies based on holographic principles: Nature, v.294, p.618-623.

76. Vanderkulk,W., Rosen,F., and Lorenz,S., 1965, Large aperture array signal processing study, IBM Final Report, DARPA Contract SD-296, 15 July 1965.

77. Van Trees,H., 1968, Detection, Estimation and Modulation theory, vol. 1: J.Wiley & Sons, New York.

78. Uski,M., 1990, Event detection and location performance of the FINESA array in Finland: Bul. Seism. Soc. Am. vol. 80, p.181- 8-1832.

79. Wielandt, E., Mitronovas,W.,1975, New developments in long period seismometery, Proc. XIV Gen. Ass. ESC, Trieste, 1974, Ed. H. Stiller, Acad.Sci.GDR, Berlin.

80. Wielandt, E., Streckeisen, G.,1982, The leaf-spring seismometer: design and performance, Bul. Seism. Soc. Am.,v.72,p.2349-2367.

81. Wielandt, E.,Steim,1986, A digital Very-broad-band seismograph , Annales Geophysical.

82. Wiggins,R.A. and Robinson,E.A., 1968, Recursive solution of the multichannel filtering problems: J. Geophys. Res., vol. 70, p.1885-189 .

SYNAPSE Science Center / Moscow IRIS Data Analysis Center

Part 2

DEVELOPMENT OF PROBLEM ORIENTED PROGRAM PACKAGE FOR SEISMIC ARRAY AND NETWORK DATA ANALYSIS

S N D A

Software Manual

Version 1.1

Moscow, May 1995

CONTENT

INTRODUCTION

A seismological program package "Seismic Network Data Analysis" (SNDA) is intended for interactive processing of multichannel seismological data recorded by seismic arrays or local regional networks. The package can be regarded as a successor of the well known SAC or NORSAR SSA (EP) packages. It comprises the most programming and graphic facilities which these packages possess and has some new features which in our opinion make it more convenient for scientific applications. These features are:

1) A wide use of a data Stack (DS) concept. The DS is a special buffer in the computer (virtual) memory containing multichannel data being analyzed during given processing session. It is the backbone of the SNDA and provides a routine data handling by means of rather rich graphic facilities (including graphic parameter measuring, comparison of traces, trace zooming, channel selection etc.) as well as by a wide set of Stack commands. The latter is a tool for multichannel data pre-processing by standard procedures (such as arithmetic operations with channel data, time windowing, filtering, Fourier transform, spectral analysis, beamforming etc.).

2) A possibility to generate a specific user problem-oriented configuration of the system on the basis of semiautomatic installation of required applications. User C or FORTRAN Seismic analysis (SA) procedures become available for execution within the framework of the SNDA in interactive and batch modes in a combination with the Stack and graphic commands.

3) An internal SNDA Job Control language (JCL) which enables user to perform complex processing sessions using any SA-procedures and commands installed in the SNDA. The language incorporates the Black Board Variables (BBV) technique. BBV are the JCL's variables which values can be assigned or read in by a user SA-procedure during its execution. The BBV implementation provides opportunities to change automatically a route of processing session depending on intermediate results.

4) An ON LINE subsystem that is an important part of the SNDA providing a real time collection of multichannel data, storing them in a Discloop (a moving time window disc buffer) and a pre-processing of data flow by the multichannel filtering, beamforming and detecting procedures.

5) An amicable user interface, which provides a reliable protection of data from wrong system manipulations by beginners and a wide set of references and diagnostic messages. The interface promotes a rather quick assimilation of the package by a new user.

The system SNDA consists of two subsystems (fig. 1) aimed to solve the different problems which one meets when handles experimental data:

The first problem is to provide an on-line collection and real time preprocessing of data. The primary example of preprocessing procedures is the detection of "useful" signals. A special part of the System - Real time subsystem (RTS) is designed to solve this problem. It interacts with a data acquisition system which provides a continuous data stream.

The second problem is to thoroughly analyze the data collected with the help of rather time consuming off-line processing routines. This problem is solved by interactive off-line part of the System.

In this manual the main attention is paid to the second part of the System (chapter 3-9). This part may be used independently (without any data acquisition system provided) for processing

data stored in disc or tape files. The files can be saved in the one of standard seismological format such as CSS, SAC or SEED. A background functioning of the Real time subsystem does not influence on the exploiting the Interactive Subsystem. The RTS processing may be paused by a special command and in this case RTS does not demand any computer recourses.

1. GENERAL PRINCIPLES OF SYSTEM DESIGN

The main problem which arises in a design of such a System, that is described in the introduction, is to provide a reliable performance of application procedures comprised in the System in conditions of different intensity of data streams and activity of system users . In the Real time subsystem (RTS) of the SNDA this problem is solved by utilizing the modern methodology of structured programming. It involves the following general approaches and facilities:

1. Up-to-down projection of a system i.e. chain breaking the whole System being designed to a set of abstract levels. As the result a clear and simple logic free from local details can be provided for each level.

2. Free asynchronous communication between different system processes based on the 'port' method. Each process should know nothing about other system processes. It should possess a single input port with a queue facility and one or several output ports. It scans the input queue, executes requested actions and sends results to the output ports.

3. Semaphore facilities to control all shared computer resources.

4. An unique system process table (SPT) in which the all process ports and their priorities are defined. An installation of new processes in the System becomes a very easy task due to this table. It is simply enough to expand the SPT.

5 Provision of a single way for modification of the System. This allows to avoid a dualism in the System. Structures of System shared memory fields, values of system parameters and so on must be defined only in macro. Modification of the macros provides the correct changes of all system programs and processes. For example, the SPT should also be contained in its own macro.

6. A global system table (GST) with the specific construction. GST provides a flexibility of System handling and possibility for easy modification of the System while including new tasks, methods, algorithms and facilities. The GST is contained in the System shared memory and comprises an arbitrary amount of named structures. A catalogue is placed at the beginning of GST. It provides a direct access to the named structures. Each structure is defined by special macro, so the GST is the set of such macros. Each individual system process uses only those macros, that it needs. Being started a user program requests from the System addresses of the named GST structures, which is needed for this program execution. An installation of a new user program bacame a very simple procedure due to the GST . It is enough to expand the GST for this purpose .

2. REAL TIME SUBSYSTEM

The Real time subsystem (RTS) is a multi-processing SUN-UNIX computer system designed to provide an acquisition and processing of experimental data in a time scale of data inflow to the computer. The list below is an example of minimal set of real time processes necessary for acquisition and preliminary processing of multichannel seismic data.

General process (GL);
Interval timer (IT);
Generator of seismic information (GN);
Control Discloop process (CDL);
Adaptation process (AD);
Detecting process (DP);
Debugging process (DG);

Any other real time data processing procedures (as filtering procedures in the frequency and space domain) can be easily installed in the RTS. The brief description of the listed processes is given below.

1. The general process (GL) provides the following functions: it initiates the computer memory; plans a control information in the memory; calls for other processes (by the UNIX "Fork" routine which creates new processes) and accepts messages concerning abnormal events in existing processes. GL is also accepts requests from a user to terminate an execution of the System or to pause the on-line processes.

2. The interval timer (IT) serves for the following purposes: the IT determines a time rate of all processes in the System using facilities of then SUN computer "Timer", excites (pushes) other system processes at the proper time moments, i.e.

- every 4 seconds (any other interval may be assigned) the IT pushes the generator of seismic information (GN);
- every 20 minutes (or any other assigned time period) the IT pushes the adaptation process (AD);
- the IT counts the data samples being received and after each 1024 data samples collected it pushes the detection process (DP).

System operations may be controlled by the IT in accordance with the real (physical) time, or can be accelerated or slowed down by special IT commands.

3. The generator of seismic information (GN) serves as an RTS interface for a data acquisition system. Being prompted by the IT messages the GN periodically reads 4 sec. array data 'packets' from a communication buffer of data acquisition system. Then the GN writes these packets to a system acquisition memory buffer (AMB) arranged in the shared memory. The AMB has two parts with equal sizes and these parts are used alternatively. At one cycle the first half of the AMB is filled by the GN with 4 sec. data packets and the another half is used by the CDL to transfer previous data packets to the Discloop. At the next cycle the halves of the AMB change their functions. To avoid the simultaneous access to the same buffer part from the GN and CDL processes the "Free-occupy" flags are supplied for every part of the AMB. If the GN (or the CDL) meets the "Occupy" value of the flag it then periodically (with 0.01 sec. interval) checks the flag state. (Note that such situation are occurred very seldom because the read-write procedures

are very fast for SUN computers). Communications between real time processes can be seen in special windows which the GL opens for each process.

4. The control Discloop process (CDL) is a special control process providing read-write operations with a Discloop. The Discloop is a direct access memory buffer arranged in a disc memory of the computer. It is a logical memory space where seismic traces are stored in the demultiplex form. Being supplied to the computer in the form of 4-sec. multiplex packets the data are written to the Discloop in demultiplex form and can be read from it with portions of any length (for one or two read operations per channel). A discloop size is defined by a length of data time interval and an amount of data channels which have to be stored. It is restricted only by a computer direct access memory available. The CDL provides the following functions in the System:

a) The CDL identifies the next events:

- messages from the GN about filling a half of the ABM with data packets;
- requests from the adaptation process to transfer a next multichannel data interval (a noise pattern selected for adaptation) from the Discloop to a shared memory AD-buffer (ADMB);
- requests from the Detection process to transfer a next multichannel data interval (a current moving window for event detection) from the Discloop to a shared memory DP-buffer (DPMB);
- requests from other asynchronous on-line processes (that could exist in the RTS) to write arbitrary sets of data from the Discloop to disc files.

b) The CDL checks requests accumulated in its input queue. If the queue is empty the CDL is switched to the waiting state. When a message from the GN is received, the CDL calculates a destination position of the current packet to be transferred in the Discloop in regard of a start time of this data packet and a value of Discloop logical start point. Physically the beginning of the Discloop file is constant (in terms of SUN-OS file system), but the logical start point is moving in time. When it achieves the right end-of-file, it turns to be zero. Thus the 'physical loop' is simulated. After a new data packet is written to the Discloop the CDL modifies the discloop start point for the next 4 seconds.

If a requests from the AD or DP are received, the CDL calculates the physical start and the end discloop positions of the information requested. Then the CDL transfers data from the Discloop to the ADMB or the DPMB (the buffers in the computer shared memory) and sends a response message to the requesting process.

If a message from any other asynchronous RTS process is received, the CDL checks a presence of requested data in the Discloop. If the Discloop contains them, the CDL transfers the information from the Discloop to an assigned file, using its own little buffer. Then the CDL send an 'OK' response message to the requesting process. If the requested interval is not contained entirely in the Discloop, the CDL returns to the process a response with a diagnostic message.

5. The adaptation process (AD) is a real time process that provides adaptation of RTS data processing procedures for current noise statistical characteristics. The adaptation is performed periodically. After requested data are transferred by the CDL from the Discloop to the ADMB the AD calculates adaptive optimal group filter (AOGF) parameters and saves them in a special OGF buffer in the computer shared memory (OGFMB). The adaptation procedure is rather time consuming one and has a law priority in the RTS.

The AD output AOGF data are used by the real time detection process (DP) for detecting wave phases of seismic events. As both asynchronies processes: the AD and DP use the same

OGFMB, a renewing the AOGF data by the AD and using them by the DP can be performed only with the help of the RTS'semaphore facilities'. The initial AOGF data are set to the OGFMB while a RTS initialization, so the DP can use them just after it starts to perform.

6. The detection process (DP) is a real time process that provides detection of signal wave phases on the basis of statistically optimal chi-square detection procedure applied to output traces of AOGF-procedures steered to a set of assigned directions. Multichannel data are sequentially processed in two minute moving windows by the set of AOGF filters created by the AD process and stored in the ADMB. Thus the set of broad band traces with whitened noise is created. Then these traces are processed by the statistically optimal detector in 4 sec. moving time windows. The detector statistic values are compared with a detection threshold. An exceeding of the threshold leads to the RTS declaration that seismic signal is detected.

At the beginning of each processing session the DP is activated by IT and generates the request to the CDL using a time of a last sample in the previous session. By this way the DP creates an overlapping of data intervals to be sequentially processed. The overlapping is needed for reliability of detection. When the DP receives data from CDL, it get access to the OGFMB and runs the detection procedure. If the latter detects a signal the DP saves an information about time intervals when the detector threshold has been exceeded by some AOGF trace in a special detection table (DT). When the processing session is ended the DP saves in the DT a time of last sample processed, frees the OGFMB semaphore and turns to the waiting state till the next IT push signal.

7. The debugging process (DG) is a tool for the system manager. The DG provides a debugging of RTS during a modification of the System. If some suspensions concerning with an abnormal functioning of the RTS are emerged the DG allows to control the RTS shared memory during processing. The DG also provides a mortal overlook of the shared memory with the purpose to find a cause of failure. The DG provides displaying the RTS memory fields using their names or addresses. It also gives the possibility to slide through the memory to the right or to the left by arbitrary steps.

3. SUBSYSTEM SNDA FOR OFF-LINE DATA ANALYSIS

3.1. Information links of SNDA with the Real time subsystem

A subsystem for seismic networks data analysis (SNDA) is a problem oriented programming shell, based on the SUN-UNIX programming facilities (such as X-window Toolkit). The main purpose of the SNDA is to support the framework and accessories for an off-line multichannel data processing. The main features of the SNDA to provide this are the following:

- Stack facilities with a set of Stack operations;
- multichannel plotting facilities;
- script performing facilities;
- facilities for installation in the SNDA of new off-line data processing procedures.

The SNDA is designed as an asynchronous dialogue process that can interact with the GL-process of the on-line Real time subsystem described above. Due to this peculiarity the SNDA provides users with a tool to control the RTS on-line data processing procedures (such as the AD or DP), in particular to modify the **'input-files'**, comprising their parameters. A user can edit the 'input-files' by any standard editor . To renew the file when the editing is finished a user have to request the special function **get semaphore** The System automatically avoids conflicts that are possible, if the file to be renewed is being read out at the same time by some on-line program. We regard also as an advantage of the SNDA (as compared with the other well-known signal processing systems) that it has close links with the RTS Discloop facilities. An exclusive feature of the SNDA is, for example, the possibility to overlook the Discloop content .

3.2. Data Stack conceptions

The main concept of the SNDA is the Data Stack concept, which had been forward by the authors of the well known seismic data processing systems such as SAC by Lawrence Livermore Laboratory and SSA (EP) by NORSAR. The Data Stack (or Stack) is a random access memory buffer where multichannel seismic traces (or other multidimensional time series) are stored in the demultiplex form. Multichannel data can be written to the Stack from the RTS Discloop or from any ordinary disc file. In the latter case a special header have to be provided ahead the data in accordance with the known standards such as SEED or CSS . We keep also in the SNDA the header that the NORSAR SSA makes use.

Multichannel data are stored in the Stack with an information about characteristics of each time series, such as symbol name of data trace, sampling interval, number of samples, initial time of trace, home position in the Stack, character string with data processing prehistory. The C-structure of a trace header is shown below.

```
struct se  /* SE TRACE ELEMENT */
{
   char     *next;     /* ADDR OF NEXT ELEMENT OR NULL - END OF LIST */
   short    channum ; /* channel number */
   unsigned short flag;
      /*   .... ...1 .... ....    trace is to be plotting
           .... .... ...1 ....    trace with zero-window after zwinon
           .... .... ..1. ....    trace with only signal after zwinon
           .... .... .... ..XX    work flags used in programs
      */
   int      time;      /*  epoch  time of  trace */
   int      ms;
   float    chndel;    /* Distance between points in seconds */
   int      chnpnt;    /* The number of samples in channel */
   char     channame[12];
   char     history[8];
   float    *trace;    /*  memory addrress of trace */
   char     *noteptr; /*  memory addrress of first note_element */
   int      notesize; /*  size of first note_element */
   float    scale;     /* KOEFFICIENT of scale during packing :
                       MAX FLOAT VALUE IS DECREASED TO 32000.0
                       OR INCREASE TO 32000.0
                     */
   float    srcdel;    /* ONLY FOR SPECTRES
                  Distance between points(sec) in source channel
                  For other traces source_chndel =0;  */
};
```

3.3. Stack commands

A new concept of the SNDA in comparison, say, with the NORSAR SSA is that we distinguish simple data handling operations from more sophisticated data processing procedures. The first are performed as Stack commands and are not needed many parameters to be adjusted. They are entered from a special command window. The latter comprise the seismic analysis procedures (SA-procedures) and used to be initiated by special menu buttons. Both the Stack commands and SA- procedures can be run also within scripts which are described below. The list of major Stack commands that are now available in the SNDA is given below

```
************ STACK READ_WRITE COMMANDS *********
readdl        readback      viewseed        readseed
savecss       readcss28     readcss30       readfloat
readsacA      readsacB      readA           savefloat
savessa       readssa       savepack        readpack
************ TRACE HANDLE COMMANDS ************
clearstack    flush          copy         move
keep          stcopy         swap
rename        set_time       set_chndel
chanon        chanoff
************ WINDOW HANDLE COMMANDS ***********
winon      winoff      winstart     win1     cut
zwinon     zwinoff
************ TRACE ARITHMETIC COMMANDS ********
add    stadd  subtract    divide    scale    shift
****COMMANDS FOR STANDARD TRACE TRANSFORMATION*****
filterB  filterC   filtCresp   filterS  whitefilt
powspec  crosssp   dirfft      invfft
************ TRACE PROCESSING COMMANDS ********
rmean     resample    sqr       sqrt       meansqr
beam      opdet       compthr   onset      snrp
************ TRACE DISPLAYING COMMANDS ********
list      plot      hardcp       XYplot      contour
3Dplot    surfer    plotspec     staconfig
************ PROCESSING COMMANDS **************
script       unix      pause
************ DECORATION COMMANDS **************
footer    vertical     notes     line      frame
```

3.4. SA procedures

Seismic data processing procedures as a rule are realized as more or less sophisticated computer programs. We call them SA-procedures (seismic analysis procedures). Each SA-procedure installed in the SNDA, have to be controlled by its **input file** comprising all parameters needed to tune the procedure for given processing session. The **input file** can be displayed at the screen before program performance and be checked or edited by an ordinary text editor.

A package of SA-procedures is subdivided in the SNDA into problem domains. Each domain contains a set of programs resolving similar data processing tasks by different methods and algorithms. An example of set of domains is shown below. It composes a specific SNDA configuration for treating the problems of nuclear underground test monitoring using data from Small Aperture Seismic Array.

GRF Adaptation & synthesis,
Group filtering,
Seismic phase detection,
ML onset time estimation,
Arrival direction estimation,
Source location,
Source type identification,
Signal simulation.

The domains are exhibited in the **SA-menu** of the SNDA. Each domain consists of several competitive procedures which are represented in a corresponding **domain submenu** and each program can be run by pushing a proper button in this submenu.

A peculiarity of the SNDA, distinguishing it from the similar data processing systems, is a special facilities which simplifies installations of new data processing procedures into the System as well as system reconfigurations. The SNDA is supplied with a special program which provides automatic compilation and assembly of the System based on the analysis of a special table stored in a *menusa* file. This table have to be composed by a user with the a help of conventional editor and has to contain the list of domains, SA-procedures forming the domains, names of corresponding programs and input files.

Source of the programs must be developed either on FORTRAN or **C** languages and placed in the **snda/for** directory. Input files for the SA-procedures must be contained in the **snda/sun4/inp** directory. The only thing a user has to do with the purpose to expand SA-procedure package is to include the names of new SA-procedures into **menusa** table and to place the procedure sources and input files into the corresponding directories. Then system program **mksnda** must be initiated. This provides SNDA reconfiguration and installation in it the new SA-procedures, which became available for SNDA user. Being run, the **mksnda** program automatically creates all changes in the main system graphic program **ssapict** and **makefile**.

The **menusa** table for the SNDA version intended for monitoring of underground nuclear explosions based on data from small aperture seismic arrays is shown below:

```
#==========  N A M E ============================  PROGRAM == INPUT_FILE
domain:     "GRF Adaptation & synthesis"
      "Noise ARMA modeling"                        _marmamo.c    marmamo
      "AOGF synthesis in freq. domain"             _armagrf.c   armagrf
      "AOGF synthesis in time domain"              _grfstd.c    grfstd

domain:     "Group filtering"
      "Optimal group filtering"                    _fpsfsa.c     fpsfsa
      "Rejection group filtering                   _rgffd.c      rjffd
      "Spatial sensitivity diagrams"               _ssd.c        ssd

domain:     "Seismic phase detection"
      "Adaptive phase detector"                    _phasedet.c   phdet
      "STA/LTA detector"                           _stalta.c     stlt

domain:     "ML onset time estimation"
      "1-component onset estimator"                _eston1.c     eston1
      "3-component onset estimator"                _eston3.c     eston3

domain:     "Arrival direction estimation"
      "Conventional wide-band f-k analysis"        _tkl.c       tkl
      "Multi-mode HR F-K analysis"                 _fkan.c      fkan
      "ARMA-model HR F-K analysis"                 _spsp.c      spsp

domain:     "Source location"
      "Source location by single array"             arloc.f     arloc
      "Estimation regional travel time on station"  estimtt.f   estimtt

domain:     "Source type identification"
      "Learnig data statistics"                    _ld.c        ld
      "Selection of classification features "      _fsel.c      fsel
```

```
      "Linear discrimination"                    _ldiscr.c    ldiscr
      "Probability error estimation"             _exam.c      exam

domain:     "Signal simulation"
      "Array body waves simulation"              _bwarsim.c   bwarsim
      "Syntetic model for array"                  arlmod.f    arlmod
      "Syntetic model for network"                netmod.f    netmod
```

Input data to be processed by a SA-procedure can be taken from the SNDA Data Stack or from ordinary disc files. For the purpose to get the input data from the Stack and to save the output traces in the Stack each SA- procedure should be provided by special routines: **writest, readst, writemx, readmx**. They can substitute the FORTRAN or **C**-language standard **read/write** statements and may be controlled by some switch parameters of the **input-file**. This creates the flexibility of data processing in the framework of the SNDA.

Data traces saved in the Stack can be displayed onto the screen and thoroughly investigated with the help of SNDA plotting facilities. The latter provide such options as choosing of channels, different modes of data scaling, data zooming in a time window (set by user) measurements of amplitude and time of any data point at the traces, evaluation of amplitude and time interval between any two points at the trace. A correlation of different traces can be investigated with the help of special option that provides a drawing of several traces along the same X-Y axes with the same X-Y scales. The drawing can be managed providing coincidence of arbitrary time moments in the traces being investigated by shifting in time of corresponding traces. All plotting commands can be initiated by pushing buttons in the SNDA Main menu, which is settled above the SNDA graphic window.

Any sequence of Stack commands and SA-procedures can be performed in the SNDA interactive mode. A command line of a special stack command window is used to enter the Stack command names and a **help** button of this window can be utilized to get information about Stack commands. The SA- procedures are initiated by pushing corresponding buttons in domain submenu of SA-menu exposed in a special SA-window.

Nevertheless the interactive mode is not so convenient for an intensive study of large amount of data. For this purpose **script** facilities are provided in the SNDA. A user can compose the named text file comprising a sequence of Stack commands and SA-procedures needed to solve his data processing problem. We call such sequence the ***script***. The script execution can be initiated by entering special command in the Stack command window. A script performance is paused after displaying the Stack traces in the SNDA graphic window each time when a *plot* command is met in the script. During these pauses a user can implement the graphic facilities available in the SNDA to investigate the data. The script execution can be proceeded by pushing the ***go*** button at the SNDA Main menu. An internal **command language** is developed in the SNDA to provide execution of more sophisticated combinations of SNDA Stack commands and SA-procedures.

4. STACK COMMAND DESCRIPTIONS

4.1. General provisions

A stack command must be typed in the Command line of the special *command window*, created by clicking the ST-button in the *System Main menu*. A general syntax of the Stack commands is the following:

operation_code [Options] [Operands]

An operation_code, options and operands are to be separated by the blanks. A set of several blanks is considered to be the single blank. An operation_code is not obliged to start from the first position of the command line. A list of the Stack commands and their detailed description can be displayed by clicking the *Help* button in the Command window (fig. 2). If a syntax error is occurred during execution of a Stack command, a diagnostic message is displayed on the screen. The message contains an explanation of error reason and its location.

Two options may exist in every Stack command:

-l option, which provides displaying a list of stack channels after a command execution;

-p option, which provides a graphic imaging of stack traces after a command execution .

Channel numbers, channel labels, integer and floating point numbers and others notations may be implemented as stack command operands.

The operand **channels** nominates a sequence of channel numbers used for the command execution. This operand should have one of the following four different forms:

(i1 i2 ... iM)	means a list of channel numbers (separated by the blanks) inserted into the brackets;
(i1-i2)	a sequence of channel numbers, defined as an interval, (with ends included) inserted into the brackets);
N	an amount of the first channels of the stack;
all	the all stack channels.

The word **filename** in a command description is used to designate a file name , the letter **k** is used to designate integer numbers while the letter **f** is used to designate real numbers. For designating a time interval, defined by its start moment and a length, the syntax construction **hh.mm.ss dd.mm.yy f** is used, where:

hh.mm.ss is hours, minutes, seconds time;

dd.mm.yy is a date of the time moment ;

f - length of an interval in (floating point) seconds.

The SNDA stack commands are described below in details. The structure of the descriptions is the same for all commands and consists of: command code and format (operands, enclosed in the square brackets are optional); command name and purpose; description of the operands; example of the implementation of the command.

4.2. Read/save commands

The read commands are designed for reading data from disc files with different standard and specific formats and placing them in the SNDA stack. If the stack is not empty, data are placed at the beginning of the stack.

readdl [-l] [-p] **hh.mm.ss dd.mm.yy k [channels]**

The command for reading data from the RTS Discloop into the Stack. If the channel list is not mentioned, then the all traces are read.

hh.mm.ss is a start time of the interval to be read;
dd.mm.yy is a date of the start point of interval to be read.
k is a length of interval to be read (in integer sec.).
channels are numbers of traces to be read;
Example: **readdl -p 21.15.46 25.01.94 90 (2-5)**
This command reads from the Discloop the 90 sec. interval of trace data with the start time 21 hour 15 minute 46 seconds of January 25, 1994. The traces with numbers 2 - 5 are read. The data are put to the Stack and are drawn into the graphic window.

readback

The command for reading data, which were automatically "backuped" by the SNDA into the special disc file. (The System backups the Stack content every time before starting a SA-procedure).

readsacB [-l] [-p] filename

The command for reading a disc file with the SAC binary format.
filename is the name of input SAC file.

readsacA [-l] [-p] filename

The command for reading a disc file with the SAC alphanumeric format.
filename is the name of input SAC file.

readssa [-l] [-p] filename [channels]

The command for reading a disc file with the NORSAR SSA ASCII format.
filename is the name of input SSA file. If the channel list is not mentioned, the all traces are read.
Example: **readssa -l hindu.form**
The command reads the all traces from the file named as '*hindu.form*' and displays the channel list on the console.

readA filename channame chandel [length]

The command for reading an ASCII file of arbitrary format without any header. File must content only one channel trace.
channame is the name to be assigned to the channel in the Stack.

chandel is the data sampling interval in seconds (float) to be assigned to the channel in the stack.

length is the size (in bytes) of file part to be read. If default, the entire file is read.

Example: **readA /data/seis/hindu nor 0.05**

The command reads the entire ASCII file with the name **/data/seis/hindu** and assigns to the Stack trace the name **nor** and the data sampling interval 0.05 sec.

readcss28 [-l] [-p] filename [STA ... STA]

The command for reading data from a disc file with the **CSS 2.8** format into the Stack.

filename is the name of input CSS file (without the extension **.wfdisc**).

STA is a station code up to 6 character length. The upper-case and lower-case letters in the station code mean the same. If STA are not mentioned the all traces are read from wfdisc.

Example:

readcss28 /data/mike/tlg/89359200057 tlg ali

The command reads the data of the stations **tlg** and **ali** from the file */data/mike/tlg/89359200057.wfdisc* with the CSS 2.8 format.

readcss30 [-l] [-p] filename [-Bbegin] [-Tintvl] [-Rresampl] [-Cchannels]

The command for reading data from a disc file with the CSS 3.0 format.

filename is the name of an input CSS file without the extension **.wfdisc**;

begin is the start time (in sec. from the beginning of the disc trace) of data interval to be read;

intvl is the length of data interval (in sec.) to be read.

channels is the list of channel numbers; every channel is supposed to have three components. By default the all traces are read (till the Stack memory is exhausted).

resampl is the resampling factor , resampling is done before writing data to the Stack.

Example: **readcss30 -p 94119010000 -B30.5 -T95.5 -R5 -C7**

The command reads the first 21 traces of the **94119010000.wfdisc** file at the time interval starting from 30.5 s with the length 95.5s. Resampling factor is equal 5. The traces read are displayed in the graphic window.

readpack filename

The command for reading data from a disc file with the special SNDA packing format.. The file is supposed to be created by the SNDA **savepack** command. The all file traces are copied into the Stack.

filename is the disc file name with the extension **.pk.**

Example: **readpack hindu.pk**

The command reads the file **hindu.pk** created by the command **savepk**

readfloat filename

The command for reading data from a disc file with the floating point format. The file is supposed to be created by the SNDA **savefloat** command. The all file traces are copied to the Stack.

filename is the disc file name with the extension **.fl.**
Example : **readfloat hindu.fl**
The command reads the file **hindu.fl** created by the SNDA **savefloat** command

viewseed filename

The command for retrieving a table of content from a disk file with the SEED format and for displaying it on the console.
filename - is the name of a disc file (it may be tar file) or the name of tapedevice .
Examples: **viewseed /home/xlm/tmp/seed_big ; viewseed /dev/nrst0 ;**
The first command retrieves a table of content from the disc file with name **/home/xlm/tmp/seed_big**, the second - from the file saved at the tape with the device name **/dev/nrst0**

readseed [-l] [-p] filename [-S(Station List)] [-C(Channel List)]

The command for reading data from a disc file with the SEED format.
Station List is the list of selected station names (separated by spaces or commas and put in the brackets. A usage of the characters **'*', '?',** and **'.'** is allowed with the results as in the standard UNIX notations. The assigned station names have to be contained in the table of file content provided by the **seedwiew** command. By default the all stations are read.
Channel List is the list of selected channels separated by spaces or commas and put in the brackets. The characters **'*', '?',** and **'.'** can be used as for station list parameter.
Examples:
readseed /home/xlm/tmp/seed_big
readseed /home/xlm/tmp/seed_big -S(chto) -C(bhn bhz lhn)
The command reads the channels **bhn, bhz, lhn** of the station **chto** *from* the SEED file */home/xlm/tmp/seed_big*

The **Save** commands are designed for copying data from the Stack to a disk file with a different standard and specific formats.

savessa filename [channels] [format]

The command for copying data from the Stack into a disc file with the NORSR SSA ASCII format.
filename is the name of output file.
format is the format of the output file; The FORTRAN notations are used for formats such as: (f10.1), (f10.3), (i6). The default format is f10.1.
Example: **savessa hindu.form**
The command writes all traces from the Stack to the disc file with the name **hindu.form** using the format f10.1.

savescc

The command for copying data from the Stack into a disc file with the CSS 2.8 format. For convenience the setting of the CSS 2.8 format parameters is made with the help of a special window, created by the command (fig. 13). A user have to type the parameter

values in named command lines of the window. The parameters to be assigned are the following::
- an **output data representation** (ASCII, binary);
- an output data format (for ASCII) or data type (for binary)such as: double 8, float 4,integer 4, integer 2);
- a **station code, channel code, name of directory containing .wfdisc-file, comments.**
After setting the all parameters a user have to push the button: "**Creating .wfdisc and waveform files**". As the result the new wfdisc and waveform files are created or existing files with the such names are updated.

savepack filename

The command for copying all traces from the Stack to a file with the FORTRAN **integer 2** format. This command is used for data compressing in the SNDA.
filename is the disc file name with the extension *.pk*.
Example: **savepack hindu.pk**

savefloat filename

The command for copying all traces from the Stack to a file with the SUN4 floating point format. This command is used in the SNDA for temporary data save when pack format is not desirable for the reason of loss of accuracy of data numerical representation.
filename is the disc file name with extension **.fl.** Command is used for archiving data
Example: **savepack hindu.fl**

4.3. Commands for manipulations with data traces

clearstack

The command for deleting all traces from Stack, including the "hidden" traces. It clears up the footers and verticals, assigned by previous stack command.

flush [-l] [-p] channels

The command for deleting the Stack traces, assigned by **channels** operand.
Example: **flush (25-50)**
The command deletes the Stack traces with numbers from 25 to 50.

keep [-l] [-p] channels

The command for deleting the Stack. traces except those assigned by the *channels* operand.
Example: **keep -p 25**
The command deletes all Stack traces except the first 25 ones and plots remaining traces in the graphic window.

copy [-l] [-p] i1 [i2]

The command for creating in the Stack a copy of a trace;

i1 is the number of channel to be copied (a channel numeration can be checked by the *list* command);
i2 is the number of channel after which the created copy is placed.
The new numeration of channels is set in the Stack after the command execution. As default the new trace is inserted into the beginning of the Stack.
Example: **copy 3 2**
The command creates the copy of 3-d channel and place it after the 2-nd channel.

stcopy [-l] [-p] n
The command for creating copies of a group of traces in the Stack. The copies of first **n** traces are placed in the beginning of the Stack.
Example: **stcopy 12**
The command creates copies of first 12 traces and places them at the beginning of the Stack.

move [-l] [-p] i1 [i2]
The command for changing a position of trace in the Stack;
i1 is the number of a trace to be moved;
i2 is the number of a trace after which the trace being moved is placed. As default the trace being moved is placed in the beginning of the Stack.
Example: **move 12 3**
The command moves the 12-th trace into position after the 3-d one.

swap [-l] [-p] i1 i2
The command for swapping positions of two traces in the Stack; **i1, i2** are the numbers of traces which positions in the Stack are exchanged.
Example: swap 12 3
The command exchanges the positions of 12-th and 3-d traces.

rename [-l] [-p] channels newname
The command for assigning a new label for the trace appointed by the **channels** operand;
newname is the new name of the trace to be renamed .
Example: **rename (7) Beam**
The command assignes the new name "**Beam**" for the 7-th trace.

set_time [-l] **channels hh.mm.ss dd.mm.yy**
The command for assigning a new starting time for traces appointed by the **channels** operand;
hh.mm.ss dd.mm.yy is the new starting time and a date for traces to be treated.
Example: **set_time all 15.45.55 23.10.93**
The command assignes the new starting time to the all traces.

set_chndel [-l] [-p] channels chnsam
The command for assigning a new sampling rate for traces appointed by the *channels* operand;

chnsam is the new data sampling interval in seconds (floating point).

chanon [-l] [-p] channels

The command for "switching on" Stack traces, appointed by the **channels** operand. All Stack traces except appointed ones become "invisible" for any SNDA Stack command and SA-procedure.

chanoff [-l] [-p]

The command for "switching off" Stack traces, chosen by a previous **chanon** command. All traces "hidden" by the previous **chanon** command become again available for any SNDA Stack command and SA-procedure.

4.4. Time window commands

winon [-l] [-p] startpoint interval

The command for setting a new time window for Stack data; After the command is executed Stack data become available for any Stack command or SA-procedure only within the new window edges. Actually the all data are remained in the Stack, so a previous time window may be restored by the command **winoff.**

startpoint is the start time of a window to be set measured in seconds (real) from the beginning of an existing time window;

interval is a length of a new time window interval in seconds (real).

Example: **winon 20 90**

The command sets the time window with the length 90 sec. starting 20sec. from the beginning of the current window.

winoff [-l] [-p]

The command for restoring a previous time window in the Stack.

winstart [-l] [-p]

The command for restoring a time window, which took place before executing all *winon* commands during current SNDA session.

win1 [-l] [-p]

The command for setting for all Stack traces a time window equal to a data time interval of the first trace in the Stack.

cut [-l] [-p] channels

The command for deleting Stack data outside a current time window.

channels are the numbers of traces to be cut.

Example: **cut all**

The command deletes data of the all Stack traces outside the current time window.

zwinon [-l] [-p] channels startpoint interval

The command for setting zero values for traces samples within an assigned time window . The real values of data samples within the window are temporary saved at the bottom of the Stack and can be restored by the command **zwinoff** .

startpoint - is the start time of the window to be set (in seconds from the beginning of a current window).

interval -is the length of window time interval (in seconds).

zwinoff [-l] [-p]

The command for restoring trace samples modified by a previous "*zwinoff*" command.

Example of implementation of **zwinon, zwinoff** commands:

```
zwinon (1-25) 40 5
chanon (1-25)
.... other commands and procedures. . .
chanoff
zwinoff
```

4.5. Trace arithmetic commands

add [-p] i1 i2

The command for summing corresponding samples of two traces with numbers **i1** and **i2.** The resulting trace replaces the first one (with number **i1**) in the Stack.

Example **add -p 2 3**

The command adds samples of the 3-d trace to corresponding samples of the 2-nd trace and places the resulting trace in the 2-nd channel of the Stack; all Stack traces are displayed in the graphic window.

stadd [-p] i1 i2 k

The command for sequential execution of the *add* command with two groups of traces. The resulting traces replace the first group.

i1 is the start number of the first group of traces with sequential numbers;

i2 is the start number of the second group of traces with sequential numbers;

k is an amount of traces in the groups.

Example: **stadd 1 26 25**

The command adds the samples of the traces with numbers 26-50 to the corresponding samples of the traces with numbers 1-25; the resulting traces replace the first 25 traces.

subtract [-p] i1 i2

The command for subtracting samples of a trace with number **i2** from corresponding samples of a trace with number **i1.** The resulting trace replaces the first one.

Example **subtract 2 3**

scale [-p] channels f

The command for scaling sample values of traces with numbers assigned by the **channels** operand. Each sample of the traces is multiplied by a factor **f** .

Example **scale 10 0.035**

The command multiplies sample values of the first 10 traces by the factor 0.035 .

divide [-p] i1 i2

The command for dividing sample values of a trace with the number **i1** by corresponding sample values of a trace with number **i2**. The resulting trace replaces the first one.
Example: **divide 3 4**

shift [-p] **channels +/- f**

The command for shifting time positions of every samples for traces assigned by the **channels** operand;
+/- means the direction of the shift (+ may be skipped);
f is the value of time shift in seconds.
Examples:
shift 5 20 The command shifts the first 5 traces to the right for 20 seconds .
shift (3) -3.75 The command shifts the 3-d trace to the left for 3.75 seconds .

4.6 Commands for standard trace transformations

filterB [-p] channels -Llow -Hhigh [-Rorder] [-Ttype] [-Ppasses]

The command for band pass filtering Stack traces by a Butterworth Filter. Filter coefficients are calculated during the command execution. The output traces are placed in the beginning of the Stack.
channels are the numbers of traces to be processed;
type is the filter type: LP-low pass; HP-high pass; BP-band pass; (default); BR-band reject
low is the low frequency of the filter band (in Hz);
high is the high frequency of the filter band (in Hz);
The **law** and **high** values must satisfy the following relations:
law < high <1/2delta, where **delta** is a trace sampling interval;
if **type**=*LP* then **low** = 0.0, **high** = cut frequency<=**1/2delta;**
if **type**=HP then **low** = cut frequency, **high** = sampling frequency/2.
order is the order of analog prototype filter (integer value not exceeding 7, recommend values are 3-5, default value is 5).
passes is the amount of filtering passes of the trace: **Passes** is integer equal to 1 or 2; 1 means the filtering of traces in the forward time direction; 2 (default) means the sequential filtering of traces in the forward and the backward directions.
Example: **filterB (3-7) -L0.5 -H4.0 -R5 -P2**
The command performs the band pass filtering of traces with numbers 3-7 by the Butterworth filter with the low frequency 0.5 Hz, high frequency 4.0, order 5. The filtering is performed with 2 passes in opposite directions.

filterS [-p] channels low1 high1 low2 high2 [slope]

The command for filtering Stack traces in two frequency bands with a frequency domain procedure. The output traces are placed at the beginning of the Stack.
channels are the numbers of traces to be processed;

low1, high1 are the low and high boundaries of the first filter frequency band;
low2, high2 are the low and high boundaries of the second filter frequency band;
This values must satisfy the following relations: **high1>low1;high2> low2;low2> high1;**
If **low2 < 0,** the second band pass is not implemented while filtering. In this case the may have arbitrary value.
slope is the parameter of slope of the filter frequency response at margins of filter bands. As default, **slope**=2 .
Example: **filterS (3-7) 0.1 0.5 1 4**
The command performs the filtering of the traces with numbers 3-7 in two frequency bands (0.1-0.5) and (1-4) Hz with the default slope coefficient equal to 2.

filterC [-p] channels fc l alpha rf

The command for a low pass filtering Stack traces by a zero phase Chebushev filter. Filter coefficients are calculated during a command execution. The output traces are placed in the beginning of the Stack.
channels are the numbers of traces to be processed.
fc is the cut frequency of a low pass band (in Hz);.
l is the one-side length of filter impulse response (without zero-lag point, *l<100*).
alpha is the relative value of slope width for the filter frequency response;
rf is the resampling factor, *rf<0.5/delta,* where *delta* is a data sampling interval (in sec.)
Example: **filterC (3-7) 0.1 30 0.12 4**
The command performs the low-pass filtering of traces with numbers 3-7. The filter cut frequency is 0.1, one side length of filter impulse response is 30, slope factor is 0.12, resampling factor is 4

filtCresp i fc l alpha.rf [ps_filename]

The Command for calculating and plotting a frequency response of low pass Chebyshev filter.
i is the number of channel to be filtered;
fc is the cut frequency of a filer band (in Hz);
l is the one-side length of filter impulse response (without zero-lag point, l<100);
alpha is the relative value of slope width for the filter frequency response;
rf is the resampling factor, **rf<0.5/delta**, where **delta** is a data sampling interval (in sec.)
ps_filename is the name of Postscript file for the plotting of the filter frequency response calculated. (The default name is **plot/snda.ps**). The implementationof the **ps-filename** operand is described in detailes in the XYplot command (see 4.8).

whitefilt channels [arord]

The command for adaptive whitening filtering Stack traces. An AR-modeling of the first trace data is used for calculating the filter coefficients. Resulting traces after filtering are placed in the beginning of the Stack.
channels are numbers of traces to be processed.

arord is the order of AR-model used for calculation of filter coefficients (**arord** <= 10).

If the parameter **arord** is present, the command calculates an autoregressive (AR) model with given order for the first trace time series. The value of order, AR-model coefficients and dispersion of residuals are delivered to the Black Board variables with the names: **sndi1**-for parameter **arord**; **sndf1-sndf11**-for AR-model coefficients. **sndf12** - for dispersion of AR residuals The whitening filtering of the assigned traces is then performed using these AR-model coefficient.

If the parameter **arord** is skipped, the order, AR-model coefficient and dispersion of residials for a whitening filtering of the chosen traces are read from Black board variables with the names: **sndi1** and **sndf1-sndf11**.

Example: **whitefilt (10) 5**

The command performs the adaptive whitening filtering of the Stack trace with number 10. Values of the filter coefficients are calculated using the AR- modelling of this trace time series. The order of AR-model is 5.

powspec channels winlen order

The command for estimating power spectra of Stac traces. Output traces containing values of power spectra are placed in the beginning of the Stack. The command calculates spectra in a current Stack time window. The command **winon** have to be performed before this command assignes a proper data time interval. The power spectra calculated are placed in the beginning of the Stack.

channels are the numbers of traces to be processed (no more then 6).

winlen is the length of correlation window (in samples). **winlen** is an integer not preceding a number of samples in the current time window;

order is the order (float) of the smooting Kayser-Bessel window. The latter is used for the power spectrum estimation.

Example: **powspec (3-7) 100 5**

The command performs the estimating of the power spectra of traces with number 3-7 in the current time window. The length of
correlation window is 100, order of smooting Kayser-Bessel window is 5.

crosssp i1 i2 winlen order [ps_filename]

The command for cross-spectral analysis of two traces with numbers **i1,i2.** The command calculates cross-spectra in a current Stack time window. The command **winon** have to be performed before this command assignes a proper data time interval. The cross spectra calculated are placed in the beginning of the Stack.

i1 i2 are the numbers of Stack traces choosen for cross-spectrum analysis.

winlen is the length of correlation window (in samples). The **Winlen** must not exceed a number of samples in the current time window;

order is the order (float) of the smooting Kayser-Bessel window. The latter is used for the power spectrum estimation.

ps_filename is the name of Postscript file for the cross-spectrum plot. (The default name is **snda.ps**). The implementationof the **ps-filename** operand is described in detailes in the **XYplot** command (see 4.8).

Example: **crosssp 11 12 100 5 Gashin.ps**

The command performs the estimating of the cross-spectra for traces with numbers 11,12 in the current time window. The length of correlation window is 100, order of smooting Kayser-Bessel window is 5. The name of postscript file, containing the cross-spectrum plot is **plot/Gashin.ps**.

dirfft [-p] i [-S]

The command for transforming a Stack trace, by direct Fast Fourier Transform. Output traces are placed in the beginning of the Stack.

i is the number of the trace to be processed;

If the parameter **-S** is skipped, the two output traces are created and placed in the Stack: the real part of Fast Fourier Transform is placed as the first trace and get the new last symbol '**R**' in the trace **history**; the imagine part of Fast Fourier Transform is placed as the second trace and get the new last symbol '**I**' in the trace **history**,

If the parameter **-S** is typed, the single output trace is created in the beginning of the Stack. It contains the module of Fast Fourier Transform and get the new last symbol '**S**' in the trace *History*

Example: **dirfft (7)**

The command calculates the direct FFT of trace 7 and stores in the Stack its real and imagine parts.

Example: **dirfft (7) -S**

The command performs the same transform but creates the single output trace containing the module of complex FFT. The trace is stored at the beginning of the Stack.

invfft [-p] i1 i2

The command for transforming two start traces by inverse Fast Fourier Transform. The traces are regarded as one complex time series created by FFT of a real time series. The resulting real time series is placed in the beginning of the StackStack.

i1, i2 are the numbers of traces to be processed. The first trace is regarded as a real part of FFT and must have the symbol '*R*' at the end of the **history**. The second trace is regarded as an imagine part of FFT and must have a symbol '**I**' at the end of the **history**.

Example: **invfft (1 2)**

The command performs the Inverse FFT of the 1-st and 2-nd traces and places the output trace in the beginning of Stack.

4.7. Trace processing commands

rmean [-p] channels

The command for removing a mean value from data of every Stack trace from a **channels** set..

channels are the numbers of traces to be processed.

Example: **rmean all**

The command calculates a mean value of a trace and subtracts it from trace sample values. These operations are performed for every trace in the Stack.

resample [-p] channels k

The command for resampling Stack traces, i.e. increasing a data sampling interval with a factor **k;**

channels are the numbers of traces to be resampled;

k- is a resampling factor.

Example: **resample (7) 4**

The command performs the resampling with factor 4 of trace 7 in the Stack. It means that only 4,8,...,40,... samples of the input trace remain in the output trace.

sqr [-p] channels

The command for replacing a trace sample values by their squares

channels are the numbers of traces to be processed.

Example: **sqr (6)**

sqrt [-p] channels

The command for replacing trace sample values by the square roots of their absolute values

channels are the numbers of traces to be processed.

Example: **sqrt (6)**

The command calculates the square roots from the absolute values of samples in trace 6 and puts them into the same trace.

meansqr channels startpoint interval [BB-name]

The command for calculating a mean square values of traces at an assigned time interval. The mean square values calculated are displayed on console and may be delivered to Black Board. If command is used in a SNDA script it can deliver calculated values to script Black Board (BB) variables with array BB-name assigned in the command.

startpoint is the time of interval left edge from the beginning of the Stack, (sec.)

interval is the time length of interval for mean square calculation (sec.).

Example: **meansqr (5-9) 40 20**

The command calculates squares of trace samples at the time interval between 40 and 60 sec. and then finds their mean value. These operations are performed for the traces with numbers 5-9

BB-name is the first letter part of names of BB variables (up to 7 letter). The BB-names must have a construction as **ccccccnnn** , where **ccccccc** is an alphabetical part of the name (up to 7 letters), **nnn** is a numerical part of a name. (up to 3 digits). The BB-names must be declared in the script before the command, which uses them. The numerical part of a group name in the declaration must begin with 1.

Example of a script fragment with the **meansquare** command:

```
. float  msq1,msq2,msq3,msq4
meansqr (5 7 21 10 ) 40 20  msq
```

```
. varlist
```

Warning: The SNDA will be aborted if not all BB-variable are defined in a script

snrp channels sp1 len1 sp2 len2 [BB-name]

The command for calculating a ratio of two mean square values for different time intervals of a trace. The results are displayed on console and may be delivered to BB-variables.

sp1 len1 sp2 len2 are the startpoints and lengths of two intervals,chosen in different (usually 'noise' and 'signal') parts of the trace (see the command **meansquare**).

Example: **snrp (5-9) 40 5 0 5**

The command calculates for every trace with number 5-9 mean square values of the trace of the intervals(40-45) and (0-5) sec. and divides the second value with on first one.

The command can be used in script with BB-variables in the same manner as the **meansqr** command.

Example of a script fragment with the *snrp* command:

```
. float  ratio1,ratio2 ratio3,ratio4
snrp (5 7 21 10) 40 20 135 10 ratio
. varlist
```

beam channels coofile azimuth ap.vel [word] [-a]

The command for composing a beam trace for assigned Stack array traces. The beam trace then can be treated by an adaptive whitening filter with purpose to increase a signal-to-noise ratio. The beam is calculated to be steered in a given azimuth and apparent velocity. The beam trace is placed in the beginning of the Stack. An averaged power of processed traces is assigned to the BB variable **sndf13**. a beam trace power is assigned to the BB variable **sndf14**.

If the optional parameter ***word*** has a value **0< word <10** the command performs the adaptive autoregressive whitening filtering of the beam trace and places a whitened trace to the Stack after the beam trace. AR-parameters of the whitening filter are delivered to the SNDA Black Board variables: **sndi1=word, sndf1-sndf11** - vector of *AR* parameters.

If **word** has the value = 0, then the command performs a whitening filtering of the beam trace with AR-parameters read from the BB variables **sndi1, sndf1 - sndf11**.

channels are the numbers of Stack traces to be processed.

coofile is the name of file with array sensor coordinates.

azimuth is the steering azimuth of beam (in degrees);

ap.vel is the steering apparent velocity of the beam (in km/sec);

word is the order of AR model for the adaptive whitening filtering of beam trace.

-a is the optional parameter. If **-a** is present, the shifted in time traces (the preliminary processing results before composing of **beam**) are placed into Stack after the beam trace (with character "**a**" in a **history**).

Example: **beam (3-7) 30 5.5 -a**

The command calculates the beam steered to the azimuth 30 degrees and apparent velocity 5.5 km/sec. using the Stack traces with numbers 3-7. The beam trace is not treated with whitening filtering. The shifted in time Stack traces N(3-7) are placed in the Stack after the beam trace.

opdet i winlen [order]

The command for performing a statistically optimal (chi-square) detection of a signal within a Stack trace **i** The trace **i** is expected to be produced by the 'command **whitefilt** . The output trace, containing values of the detector statistic, is placed in the beginning of the Stack. If the parameter **order** is skipped in the command, then **order** value is read from the Black Board variable **sndi1**, and is checked to be in the interval (1-10). If order is out of the interval, the **order** is assigned by 5.

winlen is the length of a moving time window for the calculation of detector statistic values (in samples, **winlen** > *10*order*. but **winlen** < 0.3 of a current Stack time window).

order is the order of a detector statistic. This value is expected to be the same as the *order* of preceding whitening filtering and must be in limits 1-10. Default value is *order=5*;

Example: **opdet 17 5**

The command performs the detection of the signal in Stack trace 17. The length of detector time window is 5 sec. The detector order is default value 5.

compthr channels [level]

The command for comparing sample values of every trace from a shannel set with a threshold and producing new traces maintaining only data above a threshold. Trace sample values below the threshold are replaced by the threshold value. The new traces are placed in the beginning of the Stack.

channels are the numbers of Stack traces to be processed.

level - is the value of threshold given in percents from a trace maximum value.. Default value: *level*=0.

Example: **compthr (3-7)**

The command maintains trace samples with positive values and replaces negative trace samples by zero. The traces with numbers 3-7 are processed.

onset i stwin winlen itbeg [order]

The command for estimating a signal onset time within a trace with number **i.** The output trace is a maximum likelihood function (LHF) of onset moment. It is placed in the beginning of the Stack. Parameters **stwin, winlen** and an estimated value of the onset moment are assigned to the BB variables **sndf21, sndf22,** and **sndf23** accordingly.

i is the number of Stack trace to be processed;

stwin is the start timeof the trace interval to be processed from the beginning of the Stack current window . (in sec.)

winlen is the length of trace time interval to be processed;

itbeg is the time lag from **stwin** moment to start iterative LHF calculation (in sec.).

order is the order of data autoregressive model being used for LHF calculation; **order** must be integer in the limits (1-5), default value is 3;

Example: **onset 3 40 25 2**

The command performs the signal onset time estimation within the time interval (40-65) sec. of Stack trace with number 3. The iterative LHF calculation starts at 42 sec. and up to 63 sec. The order of AR-model used is 3.

4.8. Commands for displaying traces and matrix data

list

The command for displaying a list of Stack traces on the console. The information printed (includes for every trace) a trace label, amount of trace samples, data sampling interval, symbolic string of "execution history".

plot [channels] [-x] [-y] [-t]

The command for drawing Stack traces in the SNDA graphic window. The command options are:

-x - to draw the **X**-axis in every trace graph;

-y - to set up the own **Y**-scale for each trace; scale is calculated using the absolute maximum value of trace;

-t - to draw the all traces within the Stack time window, which is active for the first channel.

XYplot filename [ps-filename] [-c]

The command for running the UNIX utility **plotxy** to display graphs of one or several functions one dimensional variable. The system creates a special graphic window for displaying the plot. The window can be quitted then by a user. The plot Postscript file with assigned (or default) name can be printed latter using a standard UNIX tool.

filename is a name of a control file containing a special script managing the execution of the **plotxy** routine. This script have to be created by a user based on the description of **plotxy** utility.

-c is the option providing the ability to display and correct the *plotxy* script file before execution.

ps-filename is the name of Postscript file, which is used instead of the default name **snda.ps**. This provides the ability to save the command output for further use. If the name of Postscript file is nentioned twice: in the command and in control file then the first name is applyed.

The current directory for SNDA performing is the directory **sun4** (see the SNDA file tree, described in the Section 8). A user is recommended to place his control and Postscript files for the SNDA graphic routines **XYplot,contour** and **3Dplot** in the subdirectory **plot** of directory **sun4**. So the name mentioned in the command is interpreted by System as a continuation to **.../snda/sun4/plot/**. If the name is mentioned in the command with a leader *slash* character then it is treated as an entire puth from the root of the file system.

Examples:

XYplot tt22.par tt22.ps

The command takes the control **plotxy** file with the name **tt22.par** from the directory **plot/** and puts the resulting postscript file into the same directory with the name **tt22.ps**.

XYplot kush/tt33.par kush/tt33.ps

The command takes the control **plotxy** file with the name **tt33.par** from the directory ***/plot/kush/*** and puts the resulting postscript file into the same directory with the name **tt33ps**.

XYplot /home/lap/data/tt44.par /home/lap/ps/tt44.ps

The command takes the control plotxy file with the name **tt44.par** from the directory **/home/lap/data** and puts the resulting postscript file into the directory ***home/lap/ps/*** with the name **tt44ps**.

contour filename [ps-filename] [-c]

The command for running the UNIX utility **contour** to display a contour map for a function of two-dimensional variable. The system creates a special graphic window for displaying the plot. The window can be quitted later by a user. The plot Postscript file with the assigned (or default) name can be printed latter using the standard UNIX tool.

filename is the name of file containing a special script managing the execution of the *contour* routine. This script have to be created by a user based on the description of *contour* utility

-c is the option providing the ability to display and correct the *contour* script file before execution.

ps-filename is the name of postscript file (default name is **plot/snda.ps**). This provides the ability to save the command output for further use.

The rule for assigning the **filename** and **ps-filename** operands is the same as for **XYplot** command.

3Dplot filename [ps-filename] [-c]

The command for running the UNIX utility **gnuplot** to display a surface for a function of two-dimensional variable. The system creates a special graphic window for displaying the plot. The window can be quitted later by a user. The plot Postscript file with the assigned (or default) name can be printed latter using the standard UNIX tool.

filename is the name of file containing a special script managing the execution of the *gnuplot* routine. This script have to be created by a user based on the description of *gnuplot* utility

-c is the option providing the ability to display and correct the *gnuplot* script file before execution.

ps-filename is the name of postscript file (default name is **plot/snda.ps**). This provides the ability to save the command output for further use.

The rule for assigning the **filename** and **ps-filename** operands is the same as for **XYplot** command.

hardcp [-Ddispose] [-Pprinter] [ps-filename]

The command for plotting a current image of SNDA main graphic window on a computer Postscript printer. The Postscript file is created which can be displayed at the screen or printed using the standard UNIX system tools.

The command options are:

dispose: equal to **v** (default); leads to the vertical disposition of image on the page equal to **h** leads to horizontal disposition of image on the page .

ps-filename is the Postscript file name; (default name is **plot/snda.ps)**. The rule for assigning the **ps-filename** operand is the same as for **XYplot** command.

printer is the computer system name of the output device . If the parameter is skipped, the image is not printed. (only the Postscrtipt file is created)

Example: **hardcp -PP1**

The command performs the printout of the current image of main graphic window on the printer with the device name **P1**. The disposition of the plot is vertical. The output postscript file name is **plot/snda.ps**.

plotspec channels [-FAGO] [-Pps_filename]

The command for plotting trace spectra, calculated by the command **powspec**

channels - are the numbers of channels to be plotted (only numbers from the interval 1-6). The traces must have the symbol *S* at the end of the *history*.

The command has the following options:

F if this symbol is present the log scale is provided for frequencies along the **X**-axis. The linear scale is used if the symbol is skipped;

A if this symbol is present the log scale is provided for the power along the **Y**-axis. The linear scale is used if the symbol is skipped;

G if this symbol is present, a grid with vertical and horizontal lines is provided on the frame;

O if this symbol is present, the all spectra are plotted in the same axes. If it is skipped an every spectrum is plotted in the separate frames.

Example: **plotspec 5 -AOGF**

The command plots 5 spectrum curves from the first five Stack traces with the logarithmic scales for both the **X** and **Y**(frequency and power) axes. All five spectrum curves are plotted in the same axes at the same frame. The grid is provided on the frame.

ps_filename is the Postscript file name (default name is **plot/snda.ps).**. The rule for assigning the **ps-filename** operand is the same as for **XYplot** command.

staconfig coofile [ps_filename]

The command for plotting a positioning of sensors (in the cortesion coordinates) for a seismic array or local seismic network. A center of coordinates coincides with the sensor having number 1.

coofile is the name of file, containing the sensor names and coordinates. Every file record must have the structure: **staname**, X , Y, Z. , where **staname** is a sensor name (maximum 8 symbols).

ps_filename is the Postscript file name (default name is **plot/snda.ps).**. The rule for assigning the **ps-filename** operand is the same as for **XYplot** command.

Example: **staconfig data/nrs.name.crd**

4.9. Special System commands

unix command

The command for temporary exiting the SNDA and entering to the computer Unix Shell. The command allows one to execute any Unix Shell instructions or user programs not installed in the SNDA.

command is the text of UNIX shell instruction or the name of user program to be executed under the Unix control.

Example: **unix cp /home/lap/tmp/* scr/kush**

script filename

The command for running a SNDA script composed by statements of the internal SNDA command language.

filename is the name of the file containing the script source; the file must be placed in the subdirectory **scr** of the SNDA current directory **sun4**.

pause

The command for pausing an execution of a script until the button "GO" in the SNDA Main menu is pushed.

4.10 Commands for graphic window decoration

A set of Stack commands is designed for inserting different explanations and symbols in arbitrary places of a current image in the SNDA graphic window. The commands are helpful, for example, for preparation figures for reports, scientific papers and other publications. The explanations and symbols can be put into the graphs of any traces; besides some captions can be written under the plot. The graphic symbols are composed with polygons and rectangles (filled and empty). The created graphic symbols and explanations are stored in a system memory and remain fixed at assigned places after regarding the image until they will be explicitly canceled. We refer below both: the graphic symbols and explanations, as **notes.**

The following operands are commonly used in the command descriptions given below:

channels are numbers of traces to be decorated by a command;

x is an **X**-coordinate (in seconds) for an initial point of a note to be inserted.

y is an **Y**-coordinate (a fraction of maximum absolute Y-value of a trace graph) for an initial point of a note to be inserted. If **y**>0, the note is put above the **X**-axis; if **y**<0, the note is put below the **X**-axis; If **y**=0, the note is put exactly on the **X**-axis;

footer number <Text>

The command for inserting a character string **<Text>** into a line with **number** 1 or 2 beneath a plot drawn in the SNDA graphic window. The maximum length of string is 45 characters. The same command is used without the operand **Text** to delete the line string, inserted by a previous command **footer .**

Examples:

footer 2 Polarization analysis

The command inserts the caption "Polarization analysis" in the second line beneath the plot.
Example: **footer 2**
The command deletes the caption, inserted by the previous command **footer**.

vertical color thickness x

The command for drawing a vertical line through all traces plotted in the graphic window. To delete all vertical lines drowned before, it is necessary to execute the command :**vertical** without any parameters.

color is a color of a vertical line: **b** - black; **r** - red; **u** - blue.
thickness is a thickness of a vertical line: **o** - ordinary; **b** - bold.
x is a **X**-Coordinate of line (in sec).
Example:
vertical r b 36
The command draws the red bold vertical line at the 36 sec. of window traces.
vertical
The command deletes the all vertical lines drawn before.

notes channels x y font <Text>

The command for inserting a character string <**Text**> into trace graphs in the SNDA main window; traces are assigned by the operand **channels**; The command **notes** with the single operand **channels** is used for deleting all notes from trace graphs.

x , y are the coordinates of the left bottom corner of first string letter;
font is the text font size: **g** - great, **m** - medium, **s** - small.

Examples:
notes (1 3 5) 30 -0.2 m P-wave arrival
The command puts the note **P-wave arrival** into traces 1,3,5. The initial point of the note has the coordinates (30.-0.2), the text font size is the medium.
notes (1 3 5)
The command deletes any notes inserted by previous commands **notes** in the trace graphs with numbers 1,3,5.

line channels color thickness x1 y1 x2 y2 ...

The command for drawing a polygons in graphs of traces assigned by the operand *channels*.
color is the color of polygon line: **b** - black; **r** - red; **u** - blue.
thickness is the polygon line thickness : **o** - ordinary; **b** - bold.
x1 y1 x2 y2 are the coordinates of the polygon corners.

Example: **line (2) r b 36 0.33 37 0.33**
The command draws the single line between points with the coordinates (36, 0.33) and (37, 0.33). The line is bold and has the red color.

frame channels color fill x1 y1 x2 y2

The command for drawing rectangles in graphs of traces assigned by the operand *channels*.

color is the color of rectangle lines : **b** - black; **r** - red; **u** - blue.

fill: is the type of rectangle : **f** - rectangle is filled e.g. its internal domain is colored; **e** - rectangle is empty e.g. its internal domain is not colored.

x1 y1 are the coordinates of the left bottom corner of rectangle .

x2 y2 are the coordinates of the right upper corner of rectangle .

Example: **frame (2) r e 26.5 -0.3 28.7 0.45**

The command draws the empty rectangle with coordinates (26.5 -0.3) (28.7 0.45). The color of rectangle lines is red.

The example of the a graphic window decorated with the commands described above is shown in fig.12.

JOB CONTROL LANGUAGE, SCRIPTS

5.1. Purposes and general requirements

The conception of scripts composed in a special language is commonly used for a design of advanced data processing packages. Scripts allow to automate data processing and thus facilitates a research and routine data analysis.

The Job Control Language (JCL) is developed in the SNDA to facilitate an implementation of SNDA Stack commands and problem oriented data processing procedures (SA-procedures). The JCL script is arbitrary sequence of Stack Commands, SA-procedures and specific JCL control Statements. Any sequence of data processing procedures can be executed in the framework of the SNDA with the help of such script.

The JCL language possesses a so-called "Black Board Variables" (BBV) facility. The BBV can be used to control a data analysis by modifying parameters of procedures or even algorithms at whole in depend on intermediate processing results. They allow to manage communication between SA-procedures during a script execution. By making computational cycles and **goto**-jumps dependent on current values of BB-variables a user can change a sequence of processing steps e.g. to execute or to skip any SA procedure or Stack command in dependent on results of previous steps.

After a JCL script is started the SNDA performs first of all the examination of script statement syntax, the verification of BB-variable specifications and the validation of correctness of their applications. All script statements are being numbered and mapped in the memory, every statement is coded using the unique 128 byte format. This provides the possibility to make jumps to labels (**goto**-jumps). A script Stack commands and SA-procedure are executed sequentially in the order of their script numbers until the operator **goto** is met. The execution of each script statements is performed after assembling its executive operation code. If a **goto** operation code is met, the new address of a statement to be processed further is calculated and then this statement record is read from memory for interpretation. Before being executed, every script statement with its number is displayed on the console. If the **pause** (or **plot**) command is met the script execution is halted (for the **plot** case after displaying the Stack traces in the SNDA graphic window) until a user clicks the ***GO*** button in the Main menu. When the script execution is terminated, the message **end of script** is displayed in the console.

JCL Control Statements are the statements for declaration of BB-variables, their initialization and modification as well as for change an order of execution of SNDA commands in depend on values of BB-variables. The following JCL control statements exist now in the SNDA: *BBV declaration*, *assignment*, **print**, **label**, conditional and unconditional **goto**. Unlike the Stack commands and SA-procedures, JCL Control Statements must have the **"."**(dot) character as the initial character in the Statement string.

A length of any script Statement must not exceed 128 characters. Statement operands are to be separated by the blanks. Several sequential blanks are considered as the one blank. Empty lines can be set among script statements for the better script understanding. Any statement, starting with the "#" (hash mark) character, is considered as a comment and is skipped during an script execution.

5.2. The classification of black board variables and their declaration

The three types of BBV exist in the JCL: integer, float and character. Arrays of BBV are not now provided in the JCL. There are also the three levels of BBV: 1) global system BBV; 2) global SNDA BBV ; 3) local script BBV.

The **System BBV** are valid during an entire period of functioning of the Real time Subsystem. They are initiated automatically by the **gl** program and a user has not to declare them in script statements. Names of System BBV have the standard structure and are composed of three parts:

sys<T><N> where: **T** is the type of variable: **i**- integer, **f**- float, **c** - character;
N is a number of variable : 1,2, ..., 11,12,... ; For example they can be: **sysi3, sysf3, sysc17** .

The **Snda BBV** are valid during a period of performing the Interactive Subsystem SNDA. They are initiated automatically by the **snda** program and a user has not to declare them in script statements. Names of Snda BBV have the standard structure and are composed of three parts:

snd<T><N> where: **T** and **N** are defined as for the System BBV. For example they can be: **sndi3, sndf3, sndc17** .

A maximum amount of System and SNDA BB-variables of each type are defined in macro **script.h.** Now they are assigned as 100 for float and 10 for integer or character variables. The maximum character variable length is defined now as 80 (in the same macros).

Some of Snda BBV already got a standard destination:

sndi1 is an order of AR-model (<= 10);
sndi10 is an return code from SA procedure;
sndf1-sndf11 are values of AR-model parameters for whitening filtering;
sndf12 is an dispersion of AR residuals (for whitening filtering);
sndf13 is an averaged power of traces processed (for **beam** Stack command);
sndf14 is an beam trace power (for **beam** Stack command).

Another Snda BBV names can be fastened to other important variables that are conventional in different problem domains.

The **local** script **BBV** are valid during only one execution of a given script. They must be declared in this script. A local BB-variable name can consist of small Latin letters and digits. Any name must begin from a letter, a name length is limited by 10 characters, for example, **fd123, result, condition, s, i, channels, shift** . Local BBV can be declared in arbitrary lines of script before statements which use them. This have to be done as in C-program, the difference is that " ; " (semicolon) is not obligatory to be at the end of string. Script BB-variables can be initialized simultaneously with their declaration. Several variables can be listed in one declaration statement. A declaration statement must start with a variable type notation: **int, float, char** .

A value of character variable is assigned by a character string enclosed in inverted commas (as in the C-language). For example the following declaration statements are correct:

```
.int    a, b=2, cc,  k17=4661780;
.float  ddd, pi=3.145926525, fff;
.char   gg[16], hhh[] = "rmean",tcom[44];
```

In script statements any Stack commands and SA procedures can use a BBV-variables as operands. In this case any BBV names has to be preceded by the character **&**. This provides a substitution of the BBV by its current value during the interpretation of the script Statement before its execution. The construction: **&scale &channels &coef** .are the examples of correct usage of BBV names in script stack commands and SA-procedures statements.

All operations with BBV (except **print**) are not dependent from the levels of variables.

5.3 Statements for assignment of BB-variable values

The statement for assignment a value to a BBV must have the variable name in the left part of equation and an another variable name with a sign, or a number or a simple arithmetic expression composed by two operands in the right part of equation. The operands in arithmetic expressions can be either BB-variables of different types: integer or floating or numbers. A conversion of operand types is made in the SNDA automatically. The result of arithmetic expression is converted to the type of variable, placed in the left part of the statement. If a floating point number is converted to an integer one, the nearest integer is taken. Character variables can not be used in arithmetic expressions. Some examples of the assignment statements are given below.

```
. int   a,b,c
. float k,l,s
. char  t

. a = 7              b = -3          . c = -a
. Å = k              l = a           .
. a = a + b          a = c+b         . a = c+7.65
. k = l - s        . k = a-3         . k = 7.65 - 2.145416
. a = 3.1416 * b   . a = a*b         . a = a*k
. a = 3.1416 / b   . a = a/b         . a = a/k
. t = "( 1 3 5 7)"
```

5.4. Statements for printout of BB-variables

The statements have the following structure

. syslist [Sys BBV-list] . sndlist [Snd BBV-list] . varlist [Local BBV_list]

Here the command operand is the list of BB-variables of appropriate type. If one of these statements is met during a script execution, current values of BB-variables mentioned in the statement operand are displayed on the console. If the variable list is skipped, the all BB-variables of appropriate type are printed. For example, the following statements are correct.

Example:

```
. varlist    a  c  k  l
. sndlist  sndi1  sndi2  sndi10  sndf1 sndf11 sndf12
. syslist
```

5.5. Statement label

A **label** statement is used together with a *goto*-statement to provide a jump into arbitrary part of script during a script execution. The label have to be defined in a separate line just before a command, which a script control must be transferred to . The label definition is shown below:
. ***<Name>:***
Arbitrary local BBV name may be used as a label (see 5.2). The label indication is the column following the label name (separation by blank(s) is possible but is not obligatory). Examples of label definitions are: `. cycle5:` `.repeat :` `.lab77: .`

5.6. Statement GOTO

A **goto** statement is used to provide a jump during script execution to a statement following a *label* statement with a name coinciding with the **goto** operand

An example is: `. goto repeat`. The statement cause the jump to the label: . **repeat**:

5.7. Statement IF

A statement can have one of the two following forms:
*if (*comparing equation*)* **goto-<name>**
*if (*comparing equation*)* assignment-expression
Two operands are compared in the comparing equation, so there must be a variable in the left part of equation and a variable or number in the right part of it. In the JCL the following comparing relations are used:

> great then	< less then	= equal to
>= great or equal	<= less or equal	!= not equal

If types of variables being compared are different, the automatic type conversion is made with the same rules as for the BBV assignment statements (see 5.3). It is not recommend to compare float-type variables by the equality relation. It would be better to compare the absolute difference of these variables with a number which is small enough. The right part of IF-statement has the same syntax, as the *assignment* and *goto*-statements.

Examples:

```
.float a,b
.int k,i
char sss[10]
```

```
.if (Â < a) b = 333
.if ( a >26.99) a= 444
.if (a = 444) sss = "XXXXXXX"
.if (k != 1) goto repeat
.if (k <= 7) goto cycle2
```

5.8. Statement ECHO

An **echo** statement has the following form:
. echo <arbitrary text>

The Statement execution causes printing the operand **text** on the console.

An example is: **. echo Estimation of filter parameters**

5.9. Statement END

A statement **end** can be set in arbitrary line of a script and provides an immediate termination of script execution. The statement must be typed without the preceding '.' (dot) symbol. An arbitrary text may be placed after the statement. This text is considered as a comment. If statement is absent in a script, the SNDA exits from the script mode after the execution of last statement of the script.

An example is: `end of script ahead of time`

5.10. Usage of BB-variables in script statements

BB-variables can be used as operands of Stack commands or SA-procedures included in a script. In this case BBV are replaced by their current values during the script execution . In contrast with the Script Control Statements discussed above BB-variables used as operands of script Stack commands or SA-procedures must be typed with the preceding character & (ampersand). An example of script with such BBV implementations is given below.

```
. int a, b
. float eee
. char  gg, tcom[44];

.  a = 2
.  b = 5
.  gg= "flush"
.  tcom = "copy -l -p 3 7"
.  eee  = 3.1416

swap &a &b              ====>          swap  2 5
&gg (&a - &b)           ====>          flush (2 - 5)
unix&tcom               ====>          copy -l -p 3 7
scale 2 &eee            ====>          scale 2 3.1416
```

5.11. Usage of SA-procedure calls in scripts

To execute some SA-procedure in a script one have to implement the next statement:
<program-name> [Input-file-name] [Option] where:

program-name is the name of SA-procedure in the same manner as mentioned in the **menusa** file, (placed in the directory **snda**).

Input-file-name is the name of specific input file, which is created to be used in this application of the SA-procedure. This file is used instead of the standard input file of SA-procedure mentioned in the **menusa** file (and saved in directory **sun4/inp**.). If the **input-file-name** is skipped in the statement the standard input file of SA-procedure is implemented for the script execution.

Option is the indication of necessity to check or correct the input file before the execution of SA-procedure. Two options may be chosen by implementation of the symbols **-c**

or **-s** . The option-**c** is used in the case, when the appointed input file after correction must be saved with the same name (for the purpose to keep the corrections). Option **-s** is used if the corrected input file must not be saved, so the initial input file must be preserved. If the option is skipped in the statement the input file is not displayed. It is used for the SA-procedure execution with those parameter values which have been preliminary assigned by a user. Note that for any options the **standard** input file of SA-procedure (stored in the directory **inp/**) is always preserved in its initial state even if its content is corrected for the given performance of SA-procedure.

Examples:

```
ssd /home/lap/archive/myssd.inp
ssd inp/kush/ssdxxx.inp-c
ssd /home/lap/archive/myssd.inp -s
ssd -c
ssd
```

5.12. Example of the script

```
#****SCRIPT FOR DEMONSTRATION OF WEAK EXPLOSION SIGNAL EXTRACTION***
#**********************FROM EARTHQUAKE CODA************************
#script kush/adhidexp.scr
#*********************************************************
clearstack
#  MODELING MIXTURE OF EARTHQUAKE AND EXPLOSION RECORDS
#  READ HINDU KUSH EARTHQUAKE: AZI=101.4  AP.VEL=14.8
readpack /detseis/seis/alex/data/noress/hindu.pk
footer 2 HINDUKUSH EARTHQUAKE: AZI=101.4  AP.VEL=14.8
plot all
#  READ NOVAYA ZEMLYA NUCLEAR EXPLOSION: AZI=32.9  AP.VEL=10.4
readpack /detseis/seis/alex/data/noress/nz.pk
footer 1 NOVAYA ZEMLYA NUCLEAR EXPLOSION: AZI=32.9  AP.VEL=10.4
footer 2 HINDUKUSH EARTHQUAKE: AZI=101.4  AP.VEL=14.8
plot  all -y
shift (1-25) -32
footer 2 SHIFT  NOVAYA ZEMLYA TO THE LEFT
plot  all -y
winon  0 60
cut  all
footer 2 CUT WINDOW 60 SECONDS
plot  all -y
# SCALE FACTOR IS TO BE IN LIMITS: 0.01 - 0.1
scale (1-25) 0.05
stadd 1 26 25
footer 2 SUMMARIZED TRACES
plot all -y
keep 25
footer 2 SUMMARIZED TRACES ONLY
plot  all
#*********************************************************
#  DATA FILTERING IN FR.BAND 0.-5. HZ AND RESAMPLING WITH FACT. 5
#filtCresp 1 5. 31 0.025 1
filterC all 5. 31 0.025 4
#
```

```
footer 1 MIXTURE OF HINDUKUSH ERTHK. AND NOV.ZEML. EXPL.
footer 2 MODELLED DATA AFTER LP FILTERING < 5 HZ
plot all
keep 25
#plot all
#*********************************************************
# CALCULATION OF BEAM AND SIGNAL DETECTION
#
#_armagrf1 kush/rbhidexp.inp
#_fpsfsa1
beam 25 data/noress.scrd  32.9  10.4
#whitefilt   1   5
#.sndlist sndf12
#opdet  1  51
_phasedet1
#flush 1
#plot 2 -y
#.sndf12 = 8687770
#opdet  2  51
footer 2 DETECTING EXPL.SIGNAL FROM BEAM TRACE
plot 4 -y
#  sndf31 - STARTPOINT OF SIGNAL DETECTED
#  sndf32 - LENGTH OF SIGNAL INTERVAL
.varlist  sndf31  sndf32
flush 3
#*********************************************************
# CHOOSING OF INTERVAL FOR ADAPTATION
#
.sndf31=sndf31-4
zwinon all &sndf31  &sndf32
chanon 25
vertical r o &sndf31
.sndf33=sndf31+sndf32
vertical r o  &sndf33
footer 2 DATA FOR ADAPTATION
plot 25
#*************************************************
#  INTERFERING EARTHQUAKE PARAMETERS ESTIMATION
#
# SPECTRUM ESTIMATION
powspec (12)  50  5
plotspec 1
flush 1
#winon 0  25
#powspec (12)  50  5
#plotspec 1
#flush 1
#winoff
#
# ESTIMATION OF P-WAVE ARRIVAL DIRECTION
_marmamo kush/ahidexp1.inp -c
_spsp1 kush/spsp.inp -c
#
# OPTIMAL GROUP FILTER ADAPTATION
_armagrf1 kush/rhidexp.inp -c
```

```
_fpsfsa1
footer 2 GR.FILTERING OF ADAPTAION INTERVAL, AR=  , MA=
plot 5 -y
flush 4
chanoff
zwinoff
#*****************************************************
# GROUP FITERING OF TOTAL TRACE INTERVAL
#
footer 2 DATA FOR GROUP FILTERING: TOTAL INTERVAL
plot 25
#
_fpsfsa1
footer 2 GR.FILTERING OF TOTAL INTERVAL, AR=  , MA=
plot  5 -y
#*****************************************************
# ONSET TIME ESTIMATION
#
onset 3  &sndf31  &sndf32  1
onset 5  &sndf31  &sndf32  1
vertical r o &sndf23
plot 5 -y
#*****************************************************
# SIGNAL SPECTRUM ESTIMATION
#
.sndf33=sndf31+4
.sndf34=sndf32-4
winon &sndf33 &sndf34
powspec (3 4 5 6) 50 5
winoff
plotspec (1 3) -AO
plot 4 -y
plotspec (3 4) -AO
#*****************************************************
# END OF SCRIPT
end
```

6. SNDA GRAPHIC CAPABILITIES

Multichannel data stored in the SNDA Stack can be displayed in a special SNDA graphic window (fig. 1-11). Interactive graphic facilities of the SNDA allow a user to overview trace waveforms and to manipulate them. It means that a user can interactively:

to select Stack traces for drawing in the graphic window;

to set a time window providing the zooming of the plot;

to perform measuring a time of any trace point or a time interval between two trace points;

to measure an amplitude value of any trace point or an amplitude difference between any two trace points;

to change an **Y**-scale of trace plots for better displaying weak wavelets;

to make a superposition of one trace graph over another for visual assessment of their correlation;

to perform a manual picking of seismic phases in a trace wave train: to identify manually a phase type, to evaluate phase onset times and onset polarity, to save this information in a special file for following use in an event source location procedure.

to perform an overview of content of the SNDA stack and RTS Diskloop .

Thus the SNDA graphic system comprises the main advanced facilities of the known seismological program packages such as SAC, SSA, ARS, SUNPICK etc.. A user interacts with the SNDA graphic system via a special graphic window incorporating mode control buttons.

6.1. Main menu description

A user starts to operate the SNDA graphic functions by switching on an appropriate mode by a control button in the SNDA Main menu (fig. 15). The latter is placed above the graphic window and contains a row of square buttons with mode definitions. By clicking the chosen button by the left mouse key a user switches the SNDA graphic subsystem (GS) to the corresponding mode. The button became dark till a user switches the SNDA to an another mode. The description of the SNDA main menu buttons is given below. By clicking the main menu buttons one gets the following results:

ST **Stack commands** - creates a new window for entering Stack commands (fig. 2, left.). This button is also used during a script execution for terminating the script ahead of time, (i.e. the button has the two predestinations).

GO **go on** - provides a proceeding of a script performing after a pause, caused by an execution of **plot** or **pause** script command.

SA **Seismology Analysis** - switches the SNDA to the mode for selecting and running SA-procedures (see p.6.2).

DL **Discloop** - switches the GS to the mode for selecting data time interval for Discloop overview (see p. 6.3).

CH **channels** - switches the GS to the mode for channel selecting (see p.6.4).

CL **clear** - clears the SNDA graphic window, cancels the current zooming time window previously set in the **tw** mode and returns the GS to the initial time interval for trace drawing which have been set by the last **read** stack command.

BK **back** - comes back (recursively) to a previous zooming window (see p. 6.5).

TW **time-window** - switches the GS to the mode for selecting a zooming time window (see p.6.5).

ON **Onset time estimation** - switches the GS to the mode for automated estimating a seismic phase onset time (see p.6.13).

TM **Time Measuring** - switches the GS to the mode for continuous measuring a time of cursor position at the plot and evaluating a time intervals between two chosen points at the plot. (see p.6.6).

AM **Amplitude Measuring** - switches the GS to the mode for continuous measuring an amplitude of cursor position at the trace graph and evaluating an amplitude difference between two chosen points at the trace graph. (see p.6.7).

MF **Magnify** - switches the GS to the mode for magnifying weak trace wavelets while plotting (see p.6.9);

PK **Picking** - switches the GS to the mode for interactive picking seismic phases in wavetrains and preparing an arrival file (see p. 6.12);

PO **Put on** - switches the GS to the mode providing superposition of trace graphs (see p.6.10);

<- , -> switches the SG to the mode providing overview of data, stored in the Stack or in the RTS Discloop (see p.6.11);

HC **Hard Copy** - creates a Postscript file for image currently present in the graphic window;

OP **Options setting** Switches the SNDA to the "Set options" mode (see p. 6.15).

? **Description of main menu** creates a special window providing a user with a references that help him to manipulate the SNDA graphic modes (fig. 14);

Q **Quit** creates a special warning window in which a user have to confirm his intention to terminate the SNDA session. After user's confirmation the SNDA is quitted.

6.2. Execution of SA- procedures

After the SNDA is switched to the SA mode by clicking the **SA** button, there appears a special SA-menu window displaying the names of Problem domains and SA-procedures currently installed in the SNDA. A user have to choose a problem domain in the SA-menu and click the button placed to the right of Problem domain name (by the right mouse key). This results in emerging a Problem domain submenu which allows a user to select a data processing SA-procedure. A user have then to initiate the selected procedure by clicking the left (or right) mouse key.

As the result the SA-menu disappears and the standard input file of the SA-procedure chosen to be executed is displayed in the editor screen. A user may correct this file (if necessary) and then proceed the program execution by clicking the **apply** button. To cancel the procedure execution a user have to click the **quit** button. All intermediate outputs of the SA-procedure being executed are printed on the console (which have been used for the SNDA starting). If the SNDA is aborted due to an error in the SA-procedure, a user should eliminate a cause of the error and restart the SNDA performance. Note, that in the abortion situation the Real time subsystem continues its functioning and there is not necessary to restart it.

6.3. Data extraction from RTS diskloop

After the SNDA is switched to the DL mode by clicking the **DL** button a graphic image of a Discloop content is exposed in a special window in a form of "dynamic line", supplied with a time scale(fig. 3). At this scale epoch time edges of a data time window stored in the Discloop are displayed and continuously renewed. By clicking some two points on the dynamic line a user can cut off any data interval he is interested. The selected time interval is expanded up to the entire length of the dynamic line and a new smaller time window can be cut using the updated dynamic line. A user may recurrently repeat this procedure to provide an accurate selecting a needed data time interval (up to several seconds) from a huge (may by multihour) data time window stored in the Discloop. This searching procedure do not really imply a reading data from the Discloop, so it is executed immediately. After the necessary time interval is set on the dynamic line, a user have to click a right mouse key that causes emerging a subwindow for the channel selection. By selecting the channels in this window a user accomplishes reading data from the Discloop to the Stack and plotting them in the SNDA graphic window.

6.4. Selection of Data Stack channels

For selecting the Stack traces which have to be plotted in the graphic window a special subwindow is provided in the SNDA. It has the structure of a number matrix which emerges within the graphic window button is pushed (fig. 4). Stack trace numbers are printed in a small button and any Stack trace can be selected or canceled by clicking a proper button (by the left mouse key). The buttons selected for trace exposition are highlighted by the red color. There are 10 buttons within every matrix raw, the maximal amount of the raws is 16. So, up to 160 data traces can be displayed in the graphic window. A selection of entire matrix raw (ten traces) is possible if the cursor is placed for clicking at the left raw border. Similarly a user can cancel the all raw numbers previously selected if he places the cursor at the right raw border. Besides, the two buttons **all** and

clear located above the number matrix enable a user to select (or cancel) the all traces by a single clicking of the left key.

The number matrix window also contains in the upper edge the four buttons: **X-axis ,Y-scale, Griding and Chanon.** The **X-axis** button provides drawing a horizontal **X**-axes in every trace plot. The **Y-scale** button is used to set a **Y**-scale for every trace plot in accordance with a maximum value of trace sample modules. The **Griding** button is used to set a grid composed by vertical lines in the plot. The **Chanon** button is used to "hide" the channels which are not highlighted by a user. The "hidden" traces are not available for further graphic manipulations and interactive Stack command treating, until they will be "found" by clicking again the same key.

After selecting the necessary traces and setting a desirable options a user have to click the right mouse key to initiate a trace drawing in the graphic window. Any manipulations with the cursor within the graphic window are not recommended until the trace drawing process is completed. To halt the trace drawing process a user have to click the right mouse key. This key must be clicked again to continue the drawing process.

6.5. Choosing plot zooming window

In the window mode (chosen by pushing the **TW** button) a user can set a new (shorter) time interval for displaying Stack traces using the entire length of graphic window (fig.5). Thus a zooming of trace graphs is accomplished. The selection of zooming window is made by two consequent clicking (with the left mouse key) a left and right edges of desired time interval in the graphic window.To replot the traces within the new time window (to zoom them) a user have to push the right mouse key. A user can repeat the operations described to achieve any desirable time scale for the trace drawings. He also can return to any previous zooming window by several pushing the **BK** button in the Main menu. The current number of zooming window is printed in a small frame in the left button corner of graphic window.

The possibility also exist to delete Stack data outside the current zooming time window . For this purpose a user have to click at first the left margin of the time scale in graphic window and then - to the right time scale margin. The System asks a user to confirm his intention to delete data and then the current window becomes the basic window and gets the number zero.

6.6. Measuring time moments of trace points

In the "Time Measuring" (TM) mode (switched on by pushing the **TM** button) the SNDA performs the graphical monitoring of cursor position in respect to origin of the time axis. The cursor takes a shape of vertical line segment. The time of current cursor position is printed out in the right bottom part of graphic window in the form **hh.mm.ss.mmm** (hours, minutes, seconds, milliseconds). To measure a time interval between two points of the trace it is necessary to click consequently this point with the left mouse button(fig. 6). An accuracy of graphic time measuring is defined by a "time price" of screen pixel equal to a length of time interval displayed in the graphic window divided by a number of screen pixels. An increasing of this "price" decreases the accuracy. For example, if the trace segment with length 1 sec. is screened in the graphic window with 1000 pixels, then the accuracy (resolving power) of the time measuring is 0,001 sec.

6.7. Measuring trace amplitudes

In the "Amplitude Measuring" (**AM**) mode switched on by pushing the **AM** button the SNDA performs the graphical monitoring of cursor deviations from zero in a chosen trace band. The cursor takes a shape of a horizontal line segment. The deviation of current cursor position from zero is printed in the right bottom part of graphic window in the units of **Y**-axis scale of the trace being treated. To measure an amplitude interval between any two points the trace it is necessary to click (with the left mouse key) the upper and lower trace points which were chosen for interval measurement (fig.7). An accuracy of graphic amplitude measurement is depend on the trace **Y**-axis scale. To gain the accuracy a user have to reduce the amount of traces displayed in the graphic window and to extend the graphic window in the vertical direction (with the help of ordinary window-treating operations of the OPEN WINDOWS environment). And at last, for measuring amplitudes of weak wavelets preceding a strong signal a user can implement the "Magnify" (**MF**) graphic mode described below.

6.8. Clipping spikes in data traces

The **AM** mode provides also a facility for graphical clipping extra large (defective) wavelets (spikes), caused for example by errors in an analog-to-digital converter. In order to perform the clipping a user have to place the cursor under the spike and then click it twice with the left mouse key. The SNDA asks a user to confirm the intention to make clipping. If a user gives the confirmation the defective wavelet is clipped out. It means that the value of the cursor **Y**-coordinate is assigned to trace samples which modules are greater then the module of this value. This is made for all samples in the time interval equal to cursor length.

6.9. Magnifying weak wavelets

First amplitudes of seismic phases are often significantly less than amplitudes of a main signal. So it is difficult to analyze visually the first wavelets of seismic phase if the entire trace is scaled based on the maximum amplitude of phase signal. To overcome such shortage a user can implement the Magnify (**MF**) mode. When the **MF** mode is set, a special Magnify panel emerges (fig.8). This panel contains several buttons:

Scale - for magnifying coefficient selection;

Quit - for panel canceling;

Select channels to magnify - for selection of traces to be magnified;

Unselect channels - for canceling the selection of channels.

A user have to set a desired magnifying coefficient (equal a power of 2) with the help of panel **Scale** facility. Then he have to push the button **Select channels to magnify** and by clicking (with the left mouse key) to mark those channels that are to be magnified. The magnifying coefficient is printed above the **X**-axis of selected channels. At last a user have to push the right mouse key to provide redrawing the chosen traces with new Y-scales set in according with the assigned magnifying coefficient. An example of magnifying of the second trace is shown in fig.8. The all trace values were proportionally increased, so the main signal became out of its initial band limits. However, the first phase wavelets can be observed in details. The time and amplitude

measurements can be carried out at this mode within the initial trace band taking onto account the magnifying coefficient.

6.10. Trace superposition

In the Trace Superposition mode (switched on by pushing the **PO** button) a user can redraw several traces in the same time-amplitude axis (fig. 9). This gives him the ability to evaluate visually a correlation between this traces. A trace superposition can be made without any time shift of traces as well as with a time shift of the traces providing a coincidence of some distinctive points at the traces. The specific of mode design is that each trace can be superposed only by a lower trace in the graphic window. The trace redrawing for superposition is performed with different colors against the black background. This enhances the visual distinction of traces. In each new act of trace superposition the colors are automatically changed. In order to provide the superposition with corncidentce of some distinctive points of traces a user have to click (with the left mouse key) the points he choose to be coincided. This is made at first for the point of lower trace then for the upper one. Then a user pushes the right mouse key and the target trace become covered by the new drawn colored lower trace.

6.11. Stack overviewing

This mode provides overviewing a content of the SNDA Stack or the RTS Discloop with the help of a time window sliding along the time interval of data stored in the Stack (Discloop). The data within the current time window interval are displayed in the graphic window. If a user intends to slide with the time window in the forward (backward) time direction he has to push (with the left mouse key) the main menu button with the right side (left-side) arrow. As a result the cursor takes a shape of right (left) directed arrow. To move forward in the time a user must click that trace point which must become the left edge of new time window. After that the SNDA redraws the traces in the new time window.

A user can manage paging the stack content by displaying traces at consequent time intervals with the same length (the length of current window). For this purpose he has to keep the cursor at the right border of the graphic window. Every clicking of the left mouse button then is followed by displaying the next page of stack data. It is evident, that if a user wants to slide along the time axis in the backward direction he has to undertake the similar manipulations with the left directed arrow-shape cursor.

Note, that during the Stack overview the **Y**-axis scales of traces are kept by default the same as they were set in the graphic window image before the switching to the overview mode. However the another option can be set for which the **Y**-axis scales of traces are changed automatically for every position of moving time window in accordance with the rules of SNDA graphic imaging.

6.12. Interactive identification of seismic phase parameters

This graphic mode (which is switched on by pushing the **PK** button) provides a user with facilities to perform interactive measuring of the main parameters of seismic phases in wavetrains displayed in the graphic window. The parameters evaluated are stored in the standard CSS "arrival" file and can be then used, for example, for a location of source of the seismic event which have been analyzed. The phase parameters evaluated in the **PK** mode are the following: a type of seismic phase, phase onset time, phase arrival quality, maximum amplitude of phase, sign of first arrival wavelet. After a user switches the SNDA to this mode a special window is created which facilitates the interactive analysis of seismograms displayed in the graphic window (fig.10). The window structure is developed to provide a user with ability to carry out the interactive seismogram analysis by a recurrent way: to repeatedly overview the seismic wavetrains and to correct the seismic phase parameters set at the previous steps. Three levels of phase parameter recording are designed for this purpose. At the first level preliminary values of parameters of analyzed phase are typed in a special text line placed in the upper (shadowed) part of the window. At the second level the parameter values confirmed by a user saved in an auxiliary buffer and are displayed in the bottom light part of the window. At the third level the output "arrival" file is created to save at the disc the checked and fixed phase parameters.

The upper (shadowed) part of the window contains the following controlling buttons and command lines.

Set window — The button initiates the preparation of phase parameter file. The seismograms displayed in the graphic window are fixed and those parts of the traces which are beyond the displayed time interval are deleted from the Stack. The initial time of displayed seismograms and their time length are stored and then are delivered to the arrival file.

Form arrival — The button causes the completion of arrival file preparation session. The file is formed using the auxiliary buffer content.

File — This is a command line in which a user types a name of arrival file.

Help — The button causes emerging of a window with a reference (help) information.

Quit — The button cancels the window for phase picking.

Save line — The button causes the moving of the text line with current parameter values to the auxiliary buffer and displaying it content in the down light part of window.

Replot — The button causes the redrawing of initial seismograms in the graphic window with seismic phase marks arranged in accordance with parameters stored in the auxiliary buffer.

Phase — The button provides a user with a facility to select a phase type.

Onset — The button provides a user with a facility to assess a phase quality (in respect of signal-to-noise ratio and so on).

Polarity The button provides a user with a facility to select a polarity (+/-) of phase first wavelet.

Beneath this buttons the of parameter names menu is disposed. A clicking of any menu rectangular allows a user to assess a value of corresponding parameter using the SNDA graphic facilities. Values of parameters **Station, Chan, Group** are fixed automatically by clicking (with the left mouse key) a point in selected trace band. The parameters, **Phase Onset Polarity**, are ascertained by selecting a needed value from a submenu which emerges after clicking (with right mouse key) the proper button (disposed above menu). The parameters *Time, Amplitude, Period* are estimated by the same procedures that were explained in the descriptions of *Time Measuring* and *Amplitude measuring* modes. The only difference is that measured time and amplitude values are printed now in the picking window text lines disposed beneath the menu rectangles. In accordance with the CSS standard the phase time is measured from the start point of initial seismogram which is supplied with an absolute (epoch) time. The needed accuracy of time and amplitude measurements can be provided by changing the time and amplitude scales of trace graphic images, as it was described in the previous sections.

In addition to the automated interactive manner of parameter assessment (explained above) a user can also type in parameter values in the corresponding text lines (after placing the cursor in the line area).

A user can check the correctness of the parameter picking by pushing the **Replot** button. The initial seismograms are redrawn in the graphic window and special phase marks are arranged at the estimated moments of phase onsets. Note, that a user can edit the auxiliary buffer content which is displayed in the bottom light part of picking window by the manner used in the conventional Open Windows editor.

Being sure that the phase parameters are fixed correctly a user have to push the **Form arrival** button. The named output arrival file with the phase parameters is thus created and saved in the disc.

6.13. Estimation of seismic phase onset times

This mode provides a user with interactive graphic tool to execute the program for precision automatic estimation of seismic phase onset times. Before switching on this mode a user have to display a seismogram to be processed as the first trace in the graphic window. The zooming time window should be chosen, which provides an explicit imaging of the first arrival wavelets of seismic phase and a rather long preceding interval containing the "pure" seismic noise.

After pushing the **ON** button in the Main menu a new "onset" subwindow is created, which facilitate tuning parameters of the onset estimation SA-procedure (fig. 11). A user have to fix the values of three parameters of the procedure by clicking onset window buttons and then to run the procedure by pushing the **Apply** button. The results of seismogram processing are the likelihood function of onset time (which is displayed in the graphic window above the trace treated) and the time moment in which this function attains its maximum. This moment serves as the estimate of phase onset time. Its value is marked in the graphic window by a red vertical line and is printed at the bottom of window.

6.14. Plotting graphic window images at Postscript printers

After the **HC** button is pushed, a Postscript file with the name ***snda.ps*** is created and saved in the directory ***plot/***. The file contains the Postscript representation of a current image in the graphic window. This Postscript file is automatically displayed by the UNIX **gs** command in a new opened window. In order to get a hard (printer) copy of the image a user have to execute the Stack command **hardcp [-Pprinter]** or the ordinary UNIX command:

lpr [-Pprinter] ... ***snda/sun4/plot/snda.ps,*** by entering it in a separate *cmdtool* window.

6.15. Setting SNDA mode options

After the **OP** button is pushed, a special panel is created at the screen to facilitate the option setting (fig. 14). The current SNDA version has the four mode options.

1) "**Automatic saving Stack data before starting SA-procedures**". In the **yes**-state this option provides a user with ability to restore in the Stack the data traces and processing results which can be accumulated before a crushing of the SNDA program caused by an abortion of some SA-procedure. When this option is valid the SNDA saves a current Stack content in a special disc file before every start of any SA-procedure. To get the saved data back to the Stack proceed their processing a user have to restart the SNDA and execute the Stack command **Readback**. This option extremely facilitates debugging of new SA-procedures in the framework of the SNDA.

2) "**Ignoring Plot commands in scripts**". In the **yes**-state of this option the System skips all **plot** statements in a script being executed. Let us remember, that in the default "**no**" state of this option a performance of script is paused after every **plot** statement execution. To proceed the script a user have to push the button **GO** in the Main menu.

3) "**Replace Stack by Discloop** ..." The option provides a user with the possibility to overview a content of the RTS Discloop instead of SNDA Stack in the *Overview* graphic mode. This option is available only if the RTS subsystem is switched on.

4) "**Constant Y-scale**" This option allows a user to keep the same Y-scales for traces in the graphic window during a subsequent displaying of different parts of Stack data in the **Overview** option . Those scales along the **Y**-axes are preserved which had been set just before the **Overview** option is switched on.

7. INCORPORATION OF NEW USER PROGRAMS INTO SNDA

In order to incorporate a new SA procedure in the SNDA a user have to accomplish the following actions:

1) to correct the **menusa** file by inserting in an appropriate domain menu a new line which comprises an particular name of procedure, a name of source program file and the name of procedure input file;

2) to put the procedure source file in the **.../snda/for/** directory. If inner subroutines, called by this procedure, are developed as the separate compiled program units, then they must also be put into the **.../snda/for** directory, and appropriate lines must be added to the System **makebase** file;

3) to put the procedure input file in the **snda/sun4/inp** directory;

4) to run the **mksnda** command which generates a text of the new versions of the system graphic program **ssapict.c** and a System **makefile** and then executes the last for compiling and linking the new SNDA executable module.

A source text of the SA-procedure being installed in the SNDA should be adjusted for the procedure would be effectively run in the framework of the SNDA. If the procedure treats some (multidimensional) time series it is recommended to provide delivering this data directly from the SNDA Stack. This allows users to manage preprocessing the data with the help of the SNDA Stack commands and reviewing the data traces with the help of the SNDA graphic subsystem. If processing results of the SA-procedure are time series too, it is also desirable to deliver them to the SNDA Stack. This facilitates a further treating of this data in next processing steps or an imaging of the data in the SNDA graphic window. To gain a SA-procedure being installed to provide communications with the SNDA Stack a user can include in its source the calls of the System routines **readst, checkst**, **readmx**, **writest**, **writemx**, which are described in the next section.

All numerical or alphabetical tuning parameters of a SA-procedure must be delivered to the program via an input file. When a SA-procedure is called this file is displayed in a text editor window, and this allows a user to check and correct program parameters. It is recommended to design such structure of input file that provides text explanations of parameters to be delivered to the program.

The SNDA has a facility that allows a user to manage an interaction between SA-procedures and Stack commands. This is Black Board variable facility. A value of any BB variable can be read in a SA-procedure and/or can be assigned in this procedure with the help of special System routines **bbvir** and **bbvic**. The descriptions of this routines are given in the next Section. The **sys** and **snd** BB variables should be used in the case if communication of SA-procedures are to be provided in the interactive SNDA mode. The local script BB variables are the most convenient tool to provide communication between SA-procedures and Stack commands in the framework of single script. To implement the local BBV in script processing statements a user have to declare the BBV by the special JCL statements, placed before the processing statements.

It can happen that a SA-procedure source is written using an old FORTRAN language version which has no means for dynamic allocation of program array memory. In this case it is recommended before installation of this program in the SNDA to design a special C-header. The header must serve as an interface between the SA-procedure and the SNDA and provide the dynamic allocation of the program memory only after the SA-procedure is started. Thus a large

amount of SA-procedures can be installed in the SNDA without making its performance extremely memory and time consuming.

If the source language has the means for dynamic allocation of program memory, then a programmer should use the allocation statement for mapping any large size arrays. In a case if a large amount of data have to be written from the Stack, subroutine **checkst** should be employed. It returns the size of memory needed to contain the ordered Stack traces. Then a programmer should allocate this memory and call the subroutine **readst** using the address of allocated memory as a parameter for mapping the data, being read.

Before discussing in details the communications between SA-procedures and the SNDA system environment we should note that a general tool which provides these communications is the System parameter **stack**. This parameter is an address of a some System index structure. Its value is delivered to every SA-procedure by the System program **ssapict** and it must be transmitted to all down level System subroutines which are used in the SA-program. Due to this parameter the entire set of SNDA system environment variables becomes available for System subroutines placed at any level of the SNDA program.

7.1 Preparation of C-headers for memory consuming FORTRAN programs

With the purpose to avoid a request of a huge memory for mapping all arrays, defined in all programs, a memory in the SNDA have to be allocated dynamically. For installation of SA-procedure, which is written in an old FORTRAN a programmer must compose a special small and typical C-program (C-header) which allocates dynamically the memory for SA-procedure, taking into account the dimensions and types of arrays used in the SA-program. As the result the memory mapping become valid only during the execution of a single SA-procedure and it become free after the completion of execution. The C-header program must have the same name as the corresponding FORTRAN program with additional **underline** character before first name letter. Exactly this name (with **underline**) must be mentioned in the file **menusa** and be used in script statements to call this SA-procedure.

In a case where a FORTRAN source of SA-procedure does not define large size arrays, the composing of C-header is not so needed.

A SA-procedure at whole may be written using the C-language. In this case the name of procedure must not be started with underline.

The example of beginning fragment of a FORTRAN subroutine with array definitions and the corresponding C-header program are shown below.

```
      SUBROUTINE SSD(stack,NFM,NXM,NYM,LM,IPM,
     $    PK,ARC,X,Y,T, TREJ,WKOR,FLIST, CMFRC,CUGF,CREJF,
     $    CMP,CMFR,CMS,CIMS,CRJM,CDGF,CGOLF,CLASS,CSTMFR,
     $    OUTFILEW,OUTFILEB,OUTFILER,OUTFILEC)
C
      INTEGER STACK(*),NFM,NXM,NYM,LM,IPM
      REAL PK(IPM,LM,LM)
      REAL ARC(IPM,LM,LM)
      REAL X(LM),Y(LM),T(LM),TREJ(LM),AR(1),VR(1),WKOR(IPM)
      REAL FLIST(NFM)
      COMPLEX CMFR(LM),CMFRC(LM),CUGF(LM),CREJF(LM)
```

```
      COMPLEX CMP(LM,LM),CRESF
      COMPLEX CMS(LM,LM),CIMS(LM,LM),CRJM(LM,LM)
      COMPLEX DW,DB,DR,DC,DWS,DBS,DRS,DCS,CDENOM
      COMPLEX CGOLF(LM),CLASS(LM),CDGF(LM),CSTMFR(LM)
      REAL OUTFILEW(NYM,NXM)
      REAL OUTFILEB(NYM,NXM)
      REAL OUTFILER(NYM,NXM)
      REAL OUTFILEC(NYM,NXM)
      CHARACTER*80 TEXT
      CHARACTER*40 COOFILE

/************* C-HEADER FOR SSD FORTRAN PROCEDURE   *******/
#include <stdio.h>
#include <malloc.h>

#define  NFM  3      /*  MAX QUANTATY OF FREQUENCIES BEING USED  */
#define  NXM  101    /*  MAX QUANTATY OF X-POINTS FOR SCANNING   */
#define  NYM  101    /*  MAX QUANTATY OF Y-POINTS FOR SCANNING   */
#define  LM   25     /*  MAX NUMBER OF CHANNELS BEING PROCESSED  */
#define  IPM  16     /*  MAX DEGREE OF AR AND/OR MA MODELS       */

void  _ssd(stack)    int *stack;
{
  int nfm=NFM, nxm=NXM, nym=NYM, lm=LM,ipm = IPM; int i;
  char     *PK,*ARC,*X,*Y,*T, *TREJ,*WKOR,*FLIST,*CMFRC,*CUGF,*CREJF,
           *CMP,*CMFR,*CMS,*CIMS,*CRJM,*CDGF,*CGOLF,*CLASS,*CSTMFR,
           *OUTFILEW,*OUTFILEB,*OUTFILER,*OUTFILEC;
/* ---------------------------------------------------------*/
   PK     = malloc(IPM*LM*LM * 4);   clearmem(PK);
   ARC    = malloc(IPM*LM*LM * 4);   clearmem(ARC);
   X      = malloc(LM * 4);          clearmem(X);
   Y      = malloc(LM * 4);          clearmem(Y);
   T      = malloc(LM * 4);          clearmem(T);
   TREJ   = malloc(LM * 4);          clearmem(TREJ);
   WKOR   = malloc(IPM * 4);         clearmem(WKOR);
   FLIST  = malloc(NFM * 4);         clearmem(FLIST);
   CMFRC  = malloc(LM * 8);          clearmem(CMFRC);
   CUGF   = malloc(LM * 8);          clearmem(CUGF);
   CREJF  = malloc(LM * 8);          clearmem(CREJF);
   CMP    = malloc(LM * LM * 8)  ;   clearmem(CMP);
   CMFR   = malloc(LM * 8);          clearmem(CMFR);
   CMS    = malloc(LM * LM * 8);     clearmem(CMS);
   CIMS   = malloc(LM * LM * 8);     clearmem(CIMS);
   CRJM   = malloc(LM * LM * 8);     clearmem(CRJM);
   CDGF   = malloc(LM * 8);          clearmem(CDGF);
   CGOL   = malloc(LM * 8);          clearmem(CGOLF);
   CLAS   = malloc(LM * 8);          clearmem(CLASS);
   CSTMFR= malloc(LM * 8);           clearmem(CSTMFR);
   OUTFILEW    = malloc(NYM*NXM * 4);   clearmem(OUTFILEW);
   OUTFILEB    = malloc(NYM*NXM * 4);   clearmem(OUTFILEB);
   OUTFILER    = malloc(NYM*NXM * 4);   clearmem(OUTFILER);
   OUTFILEC    = malloc(NYM*NXM * 4);   clearmem(OUTFILEC);

     ssd_(stack,&nfm,&nxm,&nym,&lm,&ipm,
          PK,ARC,X,Y,T,TREJ,WKOR,FLIST,CMFRC,CUGF,CREJF,
```

```
        CMP,CMFR,CMS,CIMS,CRJM,CDGF,CGOLF,CLASS,CSTMFR,
        OUTFILEW,OUTFILEB,OUTFILER,OUTFILEC);

   free(PK);    free(ARC);    free(X);      free(Y);      free(T);      free(TREJ);
   free(WKOR);  free(FLIST);  free(CMFRC);  free(CUGF);  free(CREJF);  free(CMP);
   free(CMFR);  free(CMS);    free(CIMS);   free(CRJM);  free(CDGF);   free(CGOLF);
   free(CLASS);free(CSTMFR);free(OUTFILEW);             free(OUTFILEB);
   free(OUTFILER);          free(OUTFILEC);
}
```

7.2. System subroutines for Stack data access.

There are two pairs of System routines, that provide the interface between user SA-programs and the SNDA Stack. The first pair: **readmx, writemx** interprets data being delivered as a FORTRAN matrix, where traces are considered as rows. A call for the subroutine **writemx** (to write a matrix to the Stack) have to be as shown below:

```
      CALL  WRITEMX (STACK,DATA,MAXNCHAN,MAXNPOINT,NCHAN,NPOINT,
     +      CHLBL,TIME, TIMEMS,SAMIN,HISTORY,FUNCSYM)
```

where the subroutine parameters have the following meanings

`INTEGER STACK•`	The system parameter, transferred from the system program **ssapict** to the SA-procedure
`REAL DATA (MAXNCHAN,MAXNPOINT)`	The data matrix, declared in the SA-program to contain the maximal amounts of channels and trace samples
`INTEGER MAXNCHAN,MAXNPOINT`	The matrix dimensions
`INTEGER NCHAN,NPOINT`	The amounts of channels and samples of multichannel data to be copied in the stack
`CHARACTER*12 CHLBL(MAXNCHAN)`	The array of names (labels) of data traces
`INTEGER TIME(MAXNCHAN)`	The array of start times of traces (in seconds from the beginning of 1970 - "epoch times")
`INTEGER TIMEMS(MAXNCHAN)`	The array of positive time corrections (in milliseconds) for epoch times
`REAL SAMIN(MAXNCHAN)`	The array of sampling intervals in every trace
`CHARACTER*8 HISTORY(MAXNCHAN)`	The Array of strings with "processing history"
`CHARACTER*2 FUNCSYM(MAXNCHAN)`	The Array of current processing symbols for every trace

A call for the **readmx** (Read from the Stack to a matrix) subroutine have to be as shown below:

```
      CALL READMX (STACK,CHANNELS,NCHAN,DATA,MAXNCHAN,MAXNPOINT,
     +           CHLBL,TIME,TIMEMS,SAMIN,CHNPNT,HISTORY)
```

where:

`INTEGER STACK(*)`	The system parameter, transferred from the system program **ssapict** to the SA-procedure
`CHAR*80 CHANNELS`	The list of the channels requested in the form of a **channels** string as in the Stack commands
`INTEGER NCHAN`	The actual numbers of traces which are read from the Stack into the SA-procedure
`REAL DATA (MAXNCHAN,MAXNPOINT)`	The data matrix, declared in the SA-program to contain the maximal amounts of channels and trace samples
`INTEGER MAXNCHAN,MAXNPOINT`	The matrix dimensions
`CHARACTER*12 CHLBL(MAXNCHAN)`	The array of names (labels) of data traces
`INTEGER TIME(MAXNCHAN)`	The array of start times of traces (in seconds from the beginning of 1970 - "epoch times"
`INTEGER TIMEMS(MAXNCHAN)`	The array of positive time corrections (in milliseconds) for epoch times
`REAL SAMIN(MAXNCHAN)`	The array of sampling intervals in every trace
`INTEGER CHNPNT(MAXNCHAN)`	The array of numbers of samples in every trace
`CHARACTER*8 HISTORY(MAXNCHAN)`	The array of "history strings"

The following FORTRAN program **testmx** is designed for testing the read-write System routines. This program may be used as a pattern for implementation of the **readmx, writemx** subroutines in SA-procedures.

```
      SUBROUTINE TESTMX (STACK)
        INTEGER STACK(*)
C ---   STACK   -  SYSTEM VARYABLE, transmitted from main C-program
      PARAMETER (MAXNCHAN=25)
        PARAMETER (MAXNPOINT=20000)
C-----------------------------------------------------------------
        REAL            DATA (MAXNCHAN,MAXNPOINT),SAMIN(MAXNCHAN)
        INTEGER*4       NCHAN,TIME(MAXNCHAN),TIMEMS(MAXNCHAN)
        INTEGER*4       CHNPNT(MAXNCHAN)
        CHARACTER*80    CHANNELS
        CHARACTER*12    CHLBL(MAXNCHAN)
        CHARACTER*8     HISTORY(MAXNCHAN)
        CHARACTER*2     FUNCSYM(MAXNCHAN)
C----------------- READ DATA FROM INPUT FILE -----------------------
      OPEN(21,FILE='inp/testmx.inp',STATUS='OLD')
      READ(21,'(A)')TEXT
      READ(21,'(A)')TEXT
      READ(21,'(A)')CHANNELS
      CLOSE(21)
```

```
        PRINT *,'string CHANNELS FROM INPUT FILE= ',CHANNELS

C------------------ READ TRACES FROM STACK INTO MATRIX DATA -----------
      CALL READMX (STACK,CHANNELS,NCHAN,DATA,MAXNCHAN,MAXNPOINT,
     +                 CHLBL,TIME,TIMEMS,SAMIN,CHNPNT,HISTORY)
C  ------           UNPUT DATA -----------------------------
C                   STACK   -  SYSTEM VARYABLE, transmitted from main C-program
C  CHARACTER*80     CHANNELS   Input string with stack number channels

C  ------           FIELDS FOR RETURNED DATA ----------------
C  INTEGER*4        NCHAN  Real amount of channels, that have been read
C  REAL             DATA (MAXNCHAN,MAXNPOINT)   Users matrix
C  INTEGER          MAXNCHAN             Dimention of matrix (channels)
C  INTEGER          MAXNPOINT            Dimention of matrix (points)
C  CHARACTER*12     CHLBL(MAXNCHAN)      Array of labels(12 bytes)
C  INTEGER*4        TIME(MAXNCHAN)       Array of epoch times in sec
C  INTEGER*4        TIMEMS(MAXNCHAN)     Array of ms
C  REAL             SAMIN(MAXNCHAN)      Array of sampling intervals in sec
C  INTEGER*4        CHNPNT(MAXNCHAN)     Array of numbers of samples
C  CHARACTER*8      HISTORY(MAXNCHAN)    Array of histories(8 bytes)
C  CHARACTER*2      FUNCSYM(MAXNCHAN)    Array of Symbol - notation of action,
C                                          to be added to history
        PRINT *,'We have read from stack NCHAN = ',NCHAN,'Channels'
        do 10 I=1,3
           PRINT *,'CHAN',I,':', (DATA(I,J),J=1,10)
10      CONTINUE
C------------------ WRITE TRACES FROM MATRIX DATA  TO STACK------------
        NPOINT = CHNPNT(1)
        CALL  WRITEMX (STACK,DATA,MAXNCHAN,MAXNPOINT,NCHAN,NPOINT,
     +        CHLBL,TIME, TIMEMS,SAMIN,HISTORY,FUNCSYM)
C  ------           ONLY UNPUT DATA ------------------------------
C                   STACK   -  SYSTEM VARYABLE, transmitted from main C-program
C  REAL             DATA (MAXNCHAN,MAXNPOINT)   Users whole matrix
C  INTEGER          MAXNCHAN             Dimention of whole matrix (channels)
C  INTEGER          MAXNPOINT            Dimention of whole matrix (points)
C  INTEGER          NCHAN    Amount of upper raws (channels) to be written
C  INTEGER          NPOINT   Amount of  points to be written
C  CHARACTER*12     CHLBL(MAXNCHAN)      Array of labels(12 bytes)
C  INTEGER*4        TIME(MAXNCHAN)       Array of epoch times in sec
C  INTEGER*4        TIMEMS(MAXNCHAN)     Array of ms
C  REAL             SAMIN(MAXNCHAN)      Array of sampling intervals in sec
C  CHARACTER*8      HISTORY(MAXNCHAN)    Array of histories(8 bytes)

        PRINT *,'We have written to stack NCHAN = ',NCHAN,'Channels'
        PRINT *,'CHECK GRAPHIC WINDOW, PLEASE '
      END
```

The another group of System routines: **writest**, **readst, checkst** interprets trace data being transferred as one dimensional array where channel data are ordered in demultiplex form. The subroutine **writest** delivers only one trace from SA-procedure into the Stack. It accepts the starting address of trace data and the number of samples to be delivered. Besides it have to get a data header's elements (as it is shown in the previous example). The new created trace is placed into the assigned position of the Stack.

A call for the subroutine **writest** (Write trace to the Stack) have to be as shown below:

```
      CALL WRITEST(STACK,DATA,CHLBL,TIME,TIMEMS,
     + CHNDEL,CHNPNT,HISTORY,FUNCSYM,POS,RC)
```

where:

INTEGER STACK(*)	The system parameter, transferred from the system program **ssapict** to the SA-procedure
REAL DATA(MAXDTLN)	The one dimensional data array
CHARACTER*12 CHNLBL	The channel label
INTEGER TIME	The start time of channel trace (in seconds from the beginning of 1970 - "epoch time")
INTEGER TIMEMS	The positive time correction (in milliseconds) for epoch time of the trace
REAL CHNDEL	The sampling interval of trace data
INTEGER CHNPNT	The amount of points in trace
CHARACTER*8 HISTORY	The history string
CHARACTER*2 FUNCSYM	The symbol of the current processing procedure
INTEGER POS	The position of new trace created in the Stack if POS=0, the trace is inserted in the beginning of stack; if POS > max Stack number, the trace is inserted on the end of the stack
INTEGER RC	Return code: means: 0 -O.K., 1 - invalid value of the parameter **pos**.

The System routine **readst** can read several traces from the Stack in accordance with a value of the string parameter **channels** (which is defined as in the Stack commands, see 4.1). Data being read are placed in the demultiplex form into a onedimensional array which have to be defined in the SA-procedure.. The subroutine returns an actual number of delivered traces and an amount of samples in every trace. Besides it returns the other elements of trace headers (trace parameters). These data are placed in corresponding arrays defined in the SA-procedure.

A call statement for the **readst** subroutine (read traces from the Stack) have to be as shown below:

```
      CALL READST (STACK,CHANNELS,MAXDTLN,REALNCHAN,DATA,CHANPOS,
     +             CHLBL,TIME,TIMEMS,CHNDEL,CHNPNT,HISTORY)
```

The three first parameters **stack channels,** and **maxdtln** of this subroutine are the input ones, values of the other parameters are returned by the subroutine .

`INTEGER STACK(*)`	The system parameter, transferred from the system program **ssapict** to the SA-procedure
`CHARACTER(80) CHANNELS`	The list of the channels requested in the form of a **channels** string as in the Stack commands
`INTEGER REALNCHAN`	The actual ctual number of channels read from the Stack into the SA-procedure
`MAXDTLN`	The maximum data length. If real amount of stack data samples exceeds this value, program **readst** gives a diagnostic message and SNDA is aborted
`REAL DATA(MAXDTLN)`	One dimensional array for the trace data
`CHARACTER*12 CHLBL(MAXNCHAN)`	The array of names (labels) of data traces
`INTEGER TIME(MAXNCHAN)`	The array of start times of traces (in seconds from the beginning of 1970 - "epoch times"
`INTEGER TIMEMS(MAXNCHAN)`	The array of positive time corrections (in milliseconds) for epoch times
`REAL SAMIN(MAXNCHAN)`	The array of sampling intervals in every trace
`INTEGER CHNPNT(MAXNCHAN)`	The array of numbers of samples in every trace
`CHARACTER*8 HISTORY(MAXNCHAN)`	The array of "history strings"

Being run the readst procedure checks the total amount of the requested data samples. If this value exceed the **maxdtln** parameter then the corresponding message is displayed onto console and SNDA is aborted.

If source language, implemented for a SA-program has the means for dynamic allocation of memory, then a programmer should use in the program **checkst** subroutine before **readst** one to get a the memory needed for Stack data mapping.

A call for the **checkst** subroutine (Read from the Stack) have to be as shown below:

```
      CALL CHECKST (STACK,CHANNELS,SUMDLEN,REALNCHAN,
     *              CHLBL,TIME,TIMEMS,CHNDEL,CHNPNT,HISTORY)
```

The two first parameters `STACK`, and `STACK, .` are the input ones, values of the all other parameters are returned by the subroutine `CHECKST .`

`INTEGER STACK(*)`	The system parameter, transferred from the system program **ssapict** to the SA-procedure
`CHARACTER(80) CHANNELS`	The list of the channels requested in the form of a **channels** string as in the Stack commands

INTEGER SUMDLEN	The returned value of the summarized number of data samples requested
INTEGER REALNCHAN	The actual ctual number of channels read from the Stack into the SA-procedure
CHARACTER*12 CHLBL(MAXNCHAN)	The array of names (labels) of data traces
INTEGER TIME(MAXNCHAN)	Array of start times of the channel traces (in seconds from the beginning of 1970 - "epoch time")
INTEGER TIMEMS(MAXNCHAN)	Array of positive time corrections (in milliseconds) for epoch times
REAL (MAXNCHAN)	Array of the channel data sampling intervals
INTEGER CHNPNT(MAXNCHAN)	Array of the numbers of samples in every trace
CHARACTER*8 HISTORY(MAXNCHAN	Array of the "history strings"

The following FORTRAN program **testread** is designed for testing the **writest, readst,** and **checkst** System subroutines. This program may be used as a pattern for an implementation of the subroutines in SA-procedures.

```
          SUBROUTINE TESTREAD(STACK,MNCH,MAXDTLN,
     *     DATA,CHANPOS,TIME,TIMEMS,CHNPNT,CHNDEL, CHLBL,HISTORY)

        INTEGER STACK(*)
C ---   STACK   -  SYSTEM VARYABLE, transmitted from main C-program
C------------------------------------------------- INTVL
C      MNCH -    MAXIMAL NUMBER OF CHANNELS
C       MAXDTLN - MAXIMAL NUMBER OF DATA
c-----------------------------------------------------------------
        CHARACTER*80    CHANNELS
        INTEGER*4       SUMDLEN
        INTEGER*4       REALNCHAN
        REAL            DATA (MAXDTLN)
        INTEGER*4       CHANPOS(MNCH)
        CHARACTER*12    CHLBL(*)
        INTEGER*4       TIME(MNCH)
        INTEGER*4       TIMEMS(MNCH)
        REAL            CHNDEL(MNCH)
        INTEGER*4       CHNPNT(MNCH)
        CHARACTER*8     HISTORY(*)
        CHARACTER*2     FUNCSYM
        INTEGER*4       POS,RC
C=================================================================================
C------------------ READ DATA FROM INPUT FILE ----------------------------
      OPEN(21,FILE='inp/testread.inp',STATUS='OLD')
      READ(21,'(A)')TEXT
```

```
      READ(21,'(A)')TEXT
      READ(21,'(A)')CHANNELS
      CLOSE(21)

C------------------ CHECK DATA IN STACK -------------------------------

        CALL CHECKST (STACK,CHANNELS,SUMDLEN,REALNCHAN,
     *                CHLBL,TIME,TIMEMS,CHNDEL,CHNPNT,HISTORY)
C
C  STACK(*)                        /*SYSTEM INTEGER ARRAY*/
c  CHARACTER*80 CHANNELS           /* Input string with channel stack numbers */
C  INTEGER    SUMDLEN              /* Sum number of data points requested*/
c  INTEGER    REALNCHAN            /* Real amount of channels */
c  CHARACTER*12   CHLBL(MNCH)      /* Array of labels */
c  INTEGER     TIME(MNCH)          /* Array of epoch times in sec from 1970 */
c  INTEGER     TIMEMS(MNCH)        /* Array of ms, added to apoch time */
c  REAL        CHNDEL(MNCH)        /* Array of sampling intervals in sec */
c  INTEGER     CHNPNT(MNCH)        /* Array of numbers of samples */
c  CHARACTER*8 HISTORY(MNCH)       /* Array of old histories */
c  CHARACTER*2 FUNCSYM    /* Symbol - notation of action, added to history */

        PRINT *,'SUM AMOUNTS OF POINTS: ',SUMDLEN

C------------------ READ TRACES FROM STACK ------------------------------

        CALL READST (STACK,CHANNELS,MAXDTLN,REALNCHAN,DATA,CHANPOS,
     *               CHLBL,TIME,TIMEMS,CHNDEL,CHNPNT,HISTORY)
        IF (REALNCHAN .EQ. 0) THEN
           PRINT *,'TESTREAD: NO TRACES: '
           RETURN
        END IF
C
C  STACK(*)                        /*SYSTEM INTEGER ARRAY*/
c  CHARACTER*80 CHANNELS           /* Input string with channel stack numbers */
C  INTEGER    MAXDTLN              /* Maximal number of data points to be read*/
c  INTEGER    REALNCHAN            /* Real amount of channels */
c  REAL       DATA (MAXDTLN)       /* array of united traces   */
c  INTEGER  CHANPOS(MNCH)   /* Init pos. of channel in DATA(counted from 1)*/
c  CHARACTER*12   CHLBL(MNCH)      /* Array of labels */
c  INTEGER     TIME(MNCH)          /* Array of epoch times in sec from 1970 */
c  INTEGER     TIMEMS(MNCH)        /* Array of ms, added to apoch time */
c  REAL        CHNDEL(MNCH)        /* Array of sampling intervals in sec */
c  INTEGER     CHNPNT(MNCH)        /* Array of numbers of samples */
c  CHARACTER*8 HISTORY(MNCH)       /* Array of old histories */
c  CHARACTER*2 FUNCSYM    /* Symbol - notation of action, added to history */
C------------------ CONTROL PRINT OF PARAMETERS AND TRACES ---------------
        PRINT *,'REALNCHAN: ',REALNCHAN
        PRINT *,'DATA: ',      (DATA(I),I=1,5)
        PRINT *,'CHANPOS: ',   (CHANPOS(I),I=1,2)
        PRINT *,'CHLBL: ',     (CHLBL(I),I=1,2)
        PRINT *,'TIME: ',      (TIME(I),I=1,2)
        PRINT *,'TIMEMS: ',    (TIMEMS(I),I=1,2)
        PRINT *,'CHNDEL: ',    (CHNDEL(I),I=1,2)
        PRINT *,'CHNPNT: ',    (CHNPNT(I),I=1,2)
        PRINT *,'HISTORY: ',   (HISTORY(I),I=1,2)
```

```
C-------------------- WRITE FIRST TWO TRACES INTO STACK -----------------

        DO  10 I =1,REALNCHAN
           FUNCSYM = 'W'
           POS = 77
           HISTORY(I) = 'his'

         CALL WRITEST(STACK,DATA((I-1)*CHNPNT(1)+1),CHLBL(I),
     *          TIME(I),TIMEMS(I),CHNDEL(I),CHNPNT(I),HISTORY(I),
     *          FUNCSYM,POS,RC)
C     STACK(*) - INTEGER SNDA ARRAY
C     DATA(MAXDTLN) - REAL ARRAY FOR TIME SERIES
C     CHNLBL - CHARACTER*12 VARIABLE -lable of channel
C     TIME - INTEGER*4 VARIABLE - epoch time in sec from 1970 of
C     time series start point
C     TIMEMS - INTEGER*4 VARIABLE -  ms, added to apoch time
C     CHNDEL - REAL VARIABLE -  sampling intervals in sec
C     CHNPNT - INTEGER*4 VARIABLE - number of channel samples
C     HISTORY - CHARACTER*8 VARIABLE - channel data processing
C     history
C     FUNCSYM - CHARACTER*2 - notation of action, added to history
C     POS - INTEGER VARIABLE - position in stack (from 1) after that
C      channel data are saved:
C       IF POS > MAX NUMBER OF STACK CHANNELS (9999) THEN TO WRITE TO
C     THE END OF STACK.
c       IF POS = 0  THEN TO WRITE TO THE BEGINNING OF STACK.
c       IF POS < 0  THEN PROGRAM FAILS.
C     RC - INTEGER VARIABLE - ,RETURN CODE   0 - OK , 1 - INVALID POS C
        IF (RC .NE. 0) THEN
           PRINT *,'WRITEST IS FAILED, ERROR POS: '
           RETURN
        END IF
10      CONTINUE
      END
```

7.3. System routines for Black Board variable access

The special tool is designed in the SNDA to provide to delivering the current values of System Black Board variables to internal variables of user programs. Vice versa, user programs can update System Black Board variables. All local script BB variables, which are used in script and Stack commands, have to be declared in the script. It is not needed if a SA-program communicates with **sys** and **snd** global variables. There are two System subroutines to link BBV and SA-procedures:

subroutine **bbvc** - for link with integer and real BB-variables;

subroutine **bbvir** - for link with character BB-variables.

The explanation how to implement these System subroutines in FORTRAN SA-procedures is given below. Note, that names of BB-variables in the FORTRAN program are to be declared as characters strings. While updating BB-variables, SA-procedure have to assign the definite values to the variables being transferred. Subroutine **bbwir** may read (write) an arbitrary amount of BB variables. A call statement for the subroutine **bbvir** (write/read integer and real BB-variables) have to be as shown below:

```
CALL BBVIR (STACK,ACT,TYPE,NAMES,VALUES,NUMBER)
```

where:

INTEGER STACK(*)	The system parameter, transferred from the system program **ssapict** to the SA-procedure
CHARACTER ACT	The symbol, assigning the operation: **R** -reading, **W** - writing
CHARACTER TYPE	The symbol, assigning the type of BB-variable: **I** - integer, **F** - real
CHARACTER*10 NAMEI()	The array of the names of variables being transferred
INTEGER/REAL VALUES()	The array of the values of variables being transferred
INTEGER NUMBER	The amount of variables

In contrast with **bbvir** the subroutine **bbvc** may read (write) only one character BB variable. A call statement for the subroutine **bbvc** (write/read character BB-variable) have to be as shown:

```
CALL BBVC (STACK,ACT,NAME,VALUE,LEN ]
```

where:

INTEGER STACK(*)	The system parameter, transferred from the system program **ssapict** to the SA-procedure
CHARACTER ACT	The symbol, assigning the operation: **R** -reading, **W** - writing
CHARACTER() NAME	The name of variable
CHARACTER() VALUE	The string value of the BB-variable for write operation and a string for acceptation of the variable value for read operation
INTEGER LEN	The length of the character BB-variable

The FORTRAN program **testbb** is designed for testing the read-write **bbvir, bbvc** system subroutines. This program may be used as a pattern for an implementation of the subroutines in SA-procedure

```
      SUBROUTINE TESTBB (STACK)

        INTEGER STACK(*)
C ---   STACK   -  SYSTEM VARYABLE, transmitted from main C-program
C----------------INEGERS BB-VARIABLES ------------------
        CHARACTER*10    NAMEI(3)
        INTEGER*4       INTS(3)
C----------------REAL BB-VARIABLES ------------------
        CHARACTER*10    NAMEF(3)
        REAL            FLOATS(3)
C----------------STRING BB-VARIABLE ------------------
        CHARACTER*24    SS11
C-------------------------------------------------------
        PRINT *,'FORTRAN TESTBB STARTED'
C----------------- READ FROM BLACK-BOARD -----------------------
        NAMEI(1) =     'aa11'
        NAMEI(2) =     'AA22'
        NAMEI(3) =     'AA33'
        CALL BBVIR (STACK,'R','I',NAMEI,INTS,3)
        PRINT *,'aa11=',INTS(1),   ' AA22=',INTS(2),   ' AA33=',INTS(3)

        NAMEF(1) =     'BB11'
        NAMEF(2) =     'BB22'
        NAMEF(3) =     'BB33'
        CALL BBVIR (STACK,'R','F',NAMEF,FLOATS,3)
        PRINT *,'BB11=',FLOATS(1),' BB22=',FLOATS(2),' BB33=',FLOATS(3)

        CALL BBVC  (STACK,'R','SS11',SS11,24)
        PRINT *,'SS11=', SS11
C----------------- WRITE TO BLACK-BOARD -------------------------
        INTS(1) = INTS(1) +70
        INTS(2) = INTS(2) +70
        INTS(3) = INTS(3) +70
        CALL BBVIR (STACK,'W','I',NAMEI,INTS,3)

        FLOATS(1) = FLOATS(1) +70
        FLOATS(2) = FLOATS(2) +70
        FLOATS(3) = FLOATS(3) +70
        CALL BBVIR (STACK,'W','F',NAMEF,FLOATS,3)

        CALL BBVC  (STACK,'W','SS11', 'XXXXXXXXXX YYYYYYYY',24)
        PRINT *,'FORTRAN TESTBB ENDED'
      END
```

7.4. System subroutine to get array configuration data.

The special System routine is developed for access to array configuration data. It reads a file which contains coordinates of array sensors, searches of requested sensors names within the file name list and returns coordinates of those sensors which have been requested.

A call statement for the subroutine **readcoo** (read sensor coordinates) have to be as shown below:

```
CALL READCOO (COOFILE, NCHAN, LABELS, XC, YC, ZC, IER)
```

where:

`CHARACTER*COOFILE()`	Name of file with names and coordinate of array sensors {staname,X,Y,Z }
`INTEGER NCHAN`	The amount of station requested
`LABELS(NCHAN)`	The input array of station requested labels
`XC(NCHAN)`	The output array of sensor X-coordinate s
`YC(NCHAN)`	The output array of sensor Y-coordinates
`ZC(NCHAN)`	The output array of sensor Z-coordinates(elevates)
`INTEGER IER`	The Return code (see example below)

The following FORTRAN program **testcoo** is designed for testing of the **readcoo** subroutine. This program may be used as a pattern for an implementation of the subroutine in SA-procedures

```
      PROGRAM TESTCOO
C  ---- TEST READCOO -----------

      PARAMETER  (MAXNCHAN = 25)
      CHARACTER*10    NAMERC(1)
      INTEGER*4       RC(1)
      CHARACTER*80 COOFILE
      INTEGER    NCHAN
      CHARACTER*12 LABELS (MAXNCHAN)
      INTEGER    XC (MAXNCHAN)
      INTEGER    YC (MAXNCHAN)
      INTEGER    ZC (MAXNCHAN)
      INTEGER    IER

      NAMERC(1) = 'sndi10'
      RC(1)     = 1

       COOFILE  = 'data/nrs.name.crd'
       NCHAN  =3
       LABELS (1) = 'NRS_A0_sz'
       LABELS (2) = 'NRS_B3_sz'
       LABELS (3) = 'NRS_C7_sz'


C------------------ READ COORDINATES FROM COOFILE ---------------

      CALL READCOO (COOFILE, NCHAN, LABELS, XC, YC, ZC, IER)
C_____________________________________________________________________
c  CHARACTER*80 COOFILE  /* THE NAME OF COORD FILE */
C  INTEGER    NCHAN      /* THE NUMBER OF LABELS */
c  CHARACTER*12 LABELS (NCHAN)   /* ARRAY OF LABELS    */
c  INTEGER    XC (NCHAN)         /* OUTPUT X_COORDS ARRAY */
```

```
c  INTEGER    YC (NCHAN)          /* OUTPUT Y_COORDS ARRAY */
c  INTEGER    ZC (NCHAN)          /* OUTPUT Z_COORDS ARRAY */
c  INTEGER    IER                 /*  RESULT CODE
c        0 - OK,
c        1 - SOME NAMES ARE NOT FOUND  ( xc  > 900000.0 km )
c        2 - COOFILE IS NOT FOUND;
c        3 - SYNTAX ERROR IN COOFILE;
C_____________________________________________________________
         PRINT *,' IER =', IER

         IF      (IER .EQ.1 ) THEN
             PRINT *,' SOME NAMES ARE NOT FOUND'
         ELSE IF (IER .EQ.2 ) THEN
             PRINT *,'- COOFILE IS NOT FOUND'
             GOTO  13
         ELSE IF (IER .EQ.3 ) THEN
             PRINT *,'- SYNTAX ERROR IN COOFILE'
             GOTO  13
         ELSE
             PRINT *,'OK! COORDINATES IS ASSIGNED'
             DO 77 I=1, NCHAN
77                PRINT *,LABELS(I), XC(I),  YC(I),  ZC(I)
         ENDIF
         RETURN
C-------------------------------------------------
C        EXIT ON ERROR
13       CALL BBVIR (STACK,'W','I',NAMERC,RC,1)
         RETURN
         END
```

8. INSTALLATION THE SNDA AT USER COMPUTERS

8.1. Supporting platforms and system libraries

The RTS-SNDA data processing system can be run at different modifications of SUN-4 - computers with the Operation System SunOS-4.1.x.

The Open Windows 3.0 and XView 3.0 software libraries have to be installed. To provide a FORTRAN software compilation the system libraries compatible with F77.sa.1.4.1 have to be supplied.

8.2. Setting of environments

For effective and convenient application of the RTS-SNDA System a user should make some complements in his SUN UNIX environment. The next lines should be included in user **.mylogin** file

```
################################  SNDA ################################
set path = (. /home/user/snda/sun4 /home/user/snda/sun4  ~/bin/$ARCH $path )
setenv SNDA_PATH  /home/user/snda/sun4
setenv SNDA_OBJ  /home/user/snda/obj
echo 'Configuring SNDA'
#######################################################################
```

In these lines the **user** is a **node** to link the SNDA. Due to this environment setting the all executable modules of the RTS-SNDA System such as **gl, mksa, mksnda, snda, ipcx** may be started in arbitrary directory (see below).

8.3. Incorporation of SNDA in user file system

The SNDA System at whole may be placed in arbitrary node of a user file system. The tree of SNDA work directories has the following structure:

```
snda
      for
      sys
            include
      obj
      sun4
            scr
            inp
            plot
            ssa
            data
      doc
```

The content of every directory from this tree is explained below:

The **snda** directory contains the basic System makefile **makebase**, the basic graphic program **ssabase.c** (in source code), the file **menusa** which defines a configuration of the SNDA SA-processing subsystem and the reference files with extension **.hlp**

The **for** directory contains the source codes of FORTRAN and C-programs, composing the SNDA Seismic Analyzing Subsystem.

The **sys** directory contains the source codes of the System C-programs , providing the SNDA graphic, stack data manipulating and script facilities.

The **include** directory contains the macroses (headers) needed for compilation of the **ssabase.c** program.

The **doc** directory contains the documentation of the SNDA System.

The **obj** directory contains the object modules of the whole RTS-SNDA system and some executable files such as **mksnda**, **mksa**,. This directory becomes the current after this two executable files is started.

The **sun4** directory contains the executable modules of the RTS-SNDA System. This directory becomes the current one after the SNDA is started independently of a directory in which the starting command was entered.

The **scr** directory contains the sources of scripts composed using the SNDA Job Control Language.

The **inp** directory contains the input files of SA-procedures installed in the SNDA.

The **plot** directory serves for the storage output postscript files, control files of graphic routines and data files to be plotted.

The **ssa** directory serves for exchange intermediate data and parameters between SA-procedures.

The **data** directory contents the conventional reference data for SA-procedures, such as sensor coordinates of different arrays and networks, godographs of media models, seismograms for testing of SA-procedures and so on.

Note that a user may allocate any input, output and intermediate data to be saved in arbitrary nodes of his file system.

The all SNDA disc files except ***.hlp** and ***.h** must have UNIX file attributes permitting a user to read them and to write in them. The files with extensions ***.hlp** and ***.h** must have only read permitting attributes.

8.4. SNDA executable files, initiated by users

In the RTS-SNDA System the five executable modules exist, which can be initiated by a user:

1) gl is the module to run the Real time subsystem; this file must be initiated before initiating the **snda** executable file.

2) snda is the module to run the SNDA System.

3) mksa is the module for compiling SA-programs (composed in languages C or FORTRAN) and forming a new version of the *snda* executable module; **mksa** is executed as an ordinary makefile and does not demand any tuning parameters.

4) mksnda is the module for generating a new configuration of the SNDA System after a correction of **menusa** file. It is also executed as an ordinary makefile and does not demand any parameters; **mksnda** causes a compilation of those SA-programs, that has been edited after last execution of the **mksa** command. So **mksnda** accomplishes the all functions of **mksa** module, but is rather more time consuming due to compilation of the **ssabase.c** program..

5) ipcx is the module for releasing computer resources, occupied by the RTS-SNDA System in case if this resources were not released automatically (for example due to crushing the SNDA or suspension of the Sun operation system).

8.5. Creating new versions of SNDA executable modules

Every time after correction of SA-program source a user must run the **mksa** module. In the result of its execution current programs are recompiled and the new **snda** executable module is created.

If the **menusa** file was corrected with the purpose of reconfiguration of the SA processing Subsystem set, a user must run the **mksnda** module. In the result of its execution the new **makefile** is composed and the system graphic C-program **ssabase.c** along with the newly installed or modified SA-programs are compiled and the new executable module **snda** is created.

After a new executable module **snda** is created the System prints on the console the prompt which offers a user to rename this module by any desirable name. This facility allows several users to create their own unique configurations of the SNDA modules, utilizing the same set of the SNDA System files, and to run these modules under control the single **gl** module being run in the background computer mode. Note that the all users must have a permission to read and write the files from the SNDA system directories.

9. STARTING AND TERMINATING SNDA

9.1. Starting SNDA

The two SUN *cmdtool* (or *console*) windows are needed to initiate the SNDA.(fig. 15). In the first window one must start the **gl** module, which runs the Real time subsystem. After **gl** is started some information appears in its window. The most important is the value of the shared memory identificator (**shmid**). Several icons for real time processes emerge in the left down corner of screen. By default the Real time subsystem is kept paused if no operand followed the **gl** command was entered. After the **gl** became ready a user have to enter the command **snda <shmid>** in the second *cmdtool* window. Here **<shmid>** is the shared memory identificator printed in the **gl** window. The SNDA graphic window appears on the screen. This indicates that the SNDA system is ready to perform.

A user may run several **snda** programs at the same computer under the control of the single **gl** process being run at this computer.

If SNDA is aborted because a program error or a user mistake a user should study the diagnostic messages, correct the program error, recompile the SNDA by **mksa** command, (if it is needed) and restart the **snda** with the previous value of the shared memory identificator. Let us emphasis that it is not needed to restart **gl** in this situation.

9.2. Terminating SNDA.

After completing a work with the SNDA a user have to click the **Q**-button on the main menu (disposed above the graphic window). The SNDA asks a user to confirm his intention to quit. After confirmation the graphic window disappears and the UNIX shell prompt appears in the **cmdtool** window where SNDA was executed. Then a user must enter the 0 (zero) in the first **cmdtool** window where the **gl** process is running. This terminates the Real time subsystem. In the result of such normal RTS-SNDA termination the all computer resources occupied by the programs are released.

If during the SNDA was aborted due to program error execution the SUN operation system was suspended for some cause, some computer resources (shared memory segments, queues or semaphores) being occupied by the SNDA may not be released. In this case a user must execute the **ipcx** program. The program is specially designed to make free all computer resources occupied by the RTS-SNDA. This may be made in arbitrary moment after aliving the computer Operation system.

LIST OF ABBREVIATIONS

AD	Adaptation process
ADMB	shared memory AD-buffer
AF	Arrival file
AMB	Acquisition memory buffer
AOGF	Adaptive optimal group filter
DP	Detection process
DPMB)	shared memory DP-buffer
DS	Data stack
DT	Detection table
GN	Generator of model seismic information
GS	Graphic subsystem
GST	Global System Table
IF	Input file
IT	Interval timer
JCL	Job Control language
LASA	Low aperture Seismic array
OBF	Optimal filters buffer
OGF	Optimal group filtering
OGFMB	Memory buffer for Optimal group filters adaptation
PE	Program executor
RTS	Real time subsystem
SA	Seismic analysis
SC	Stack command
SCW	Stack control window
SL	SNDA language
SNDA	Seismic net work data analysis
SPT	System Process Table

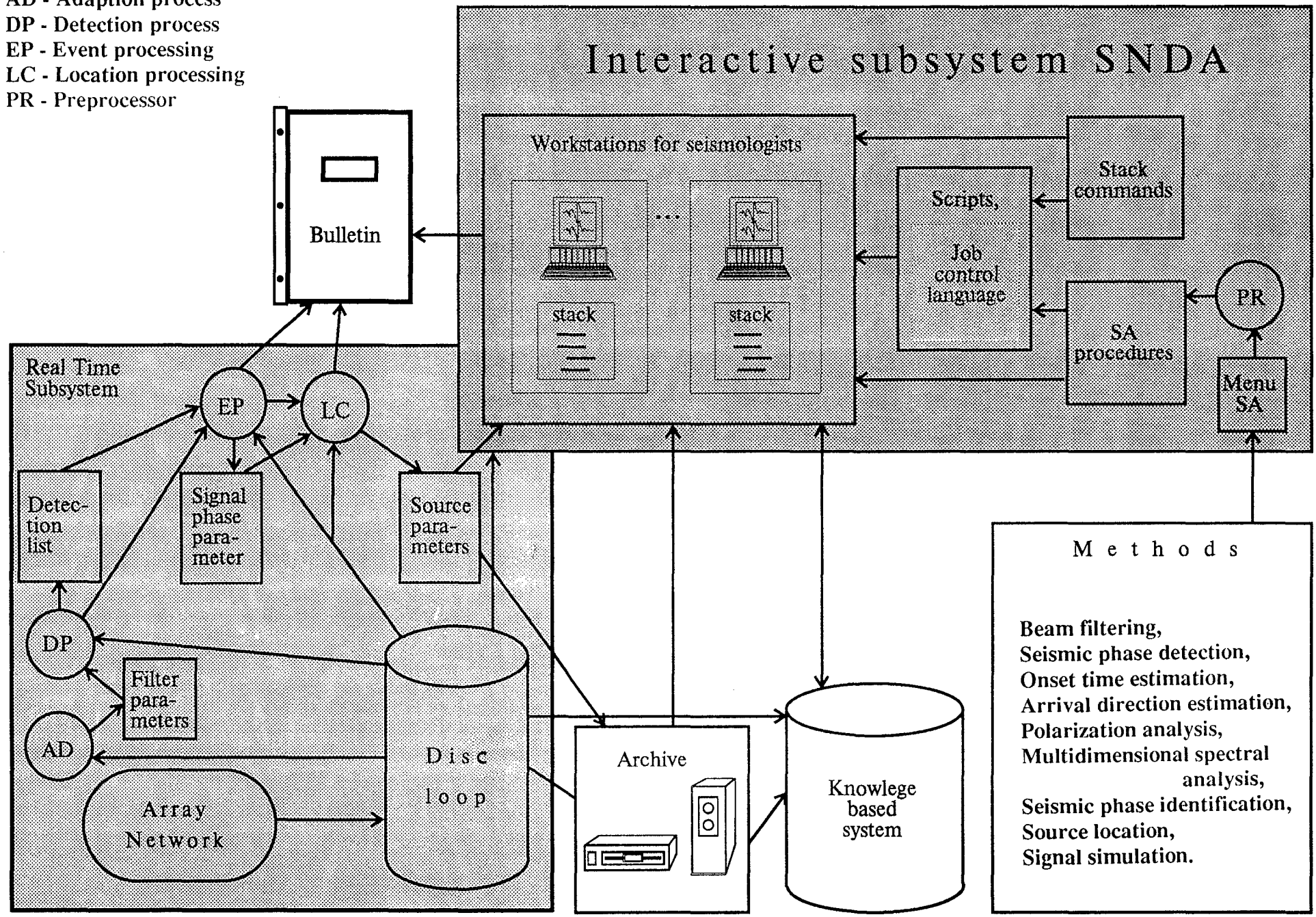


Fig.1. System information flow chart

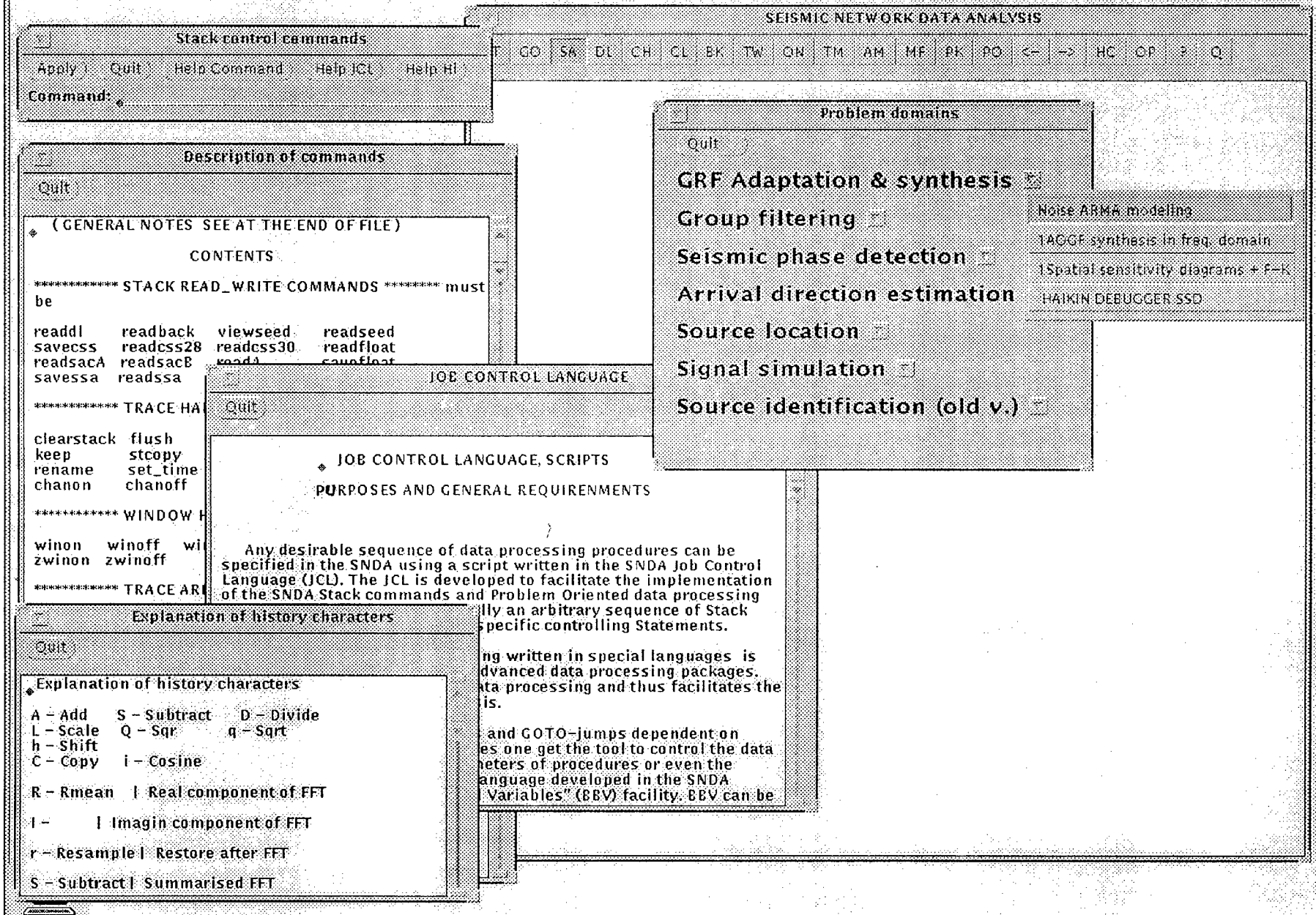


Fig. 2. Windows for stack commands and for choice of SA-procrdure

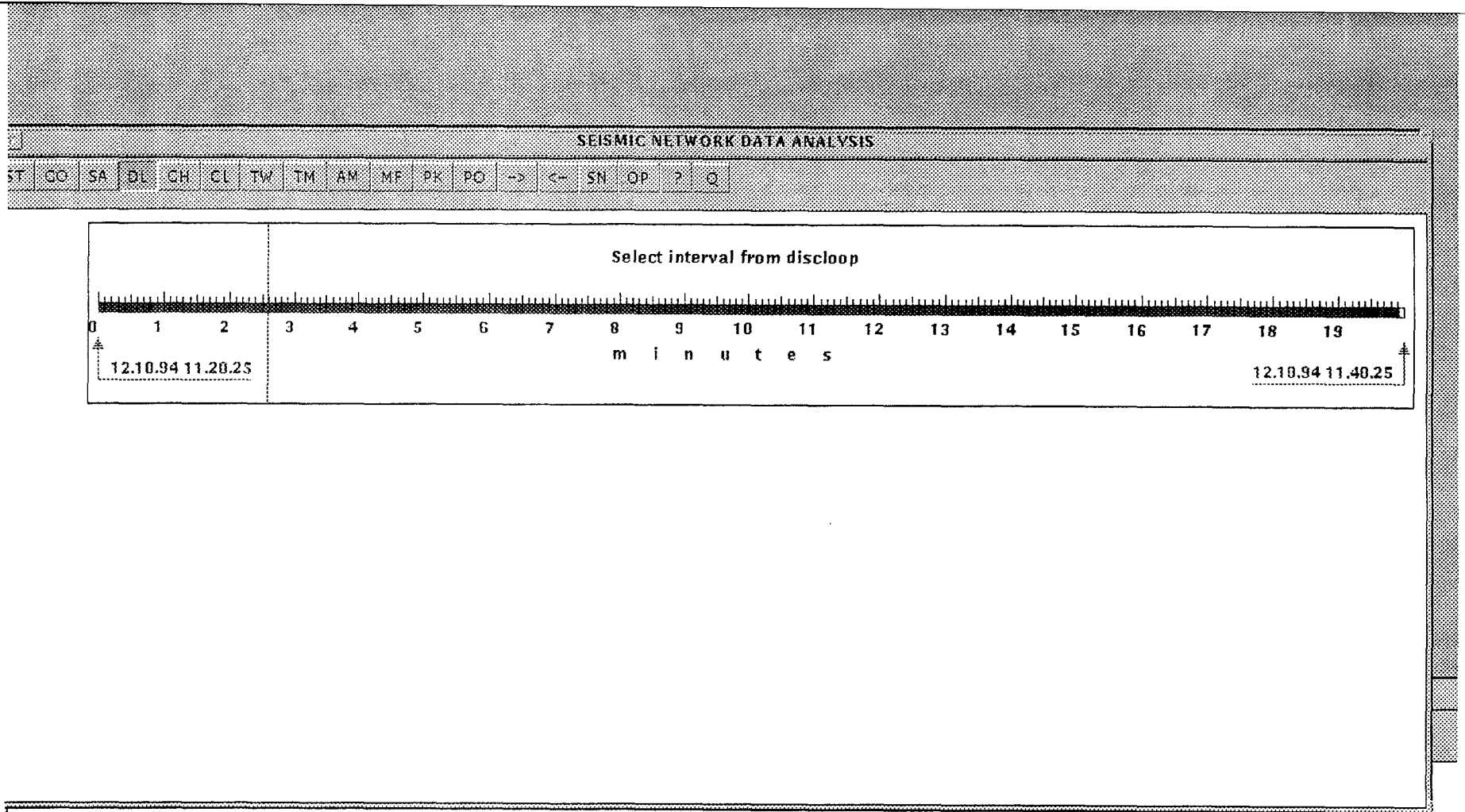


Fig. 3. Selection of interval from discloop

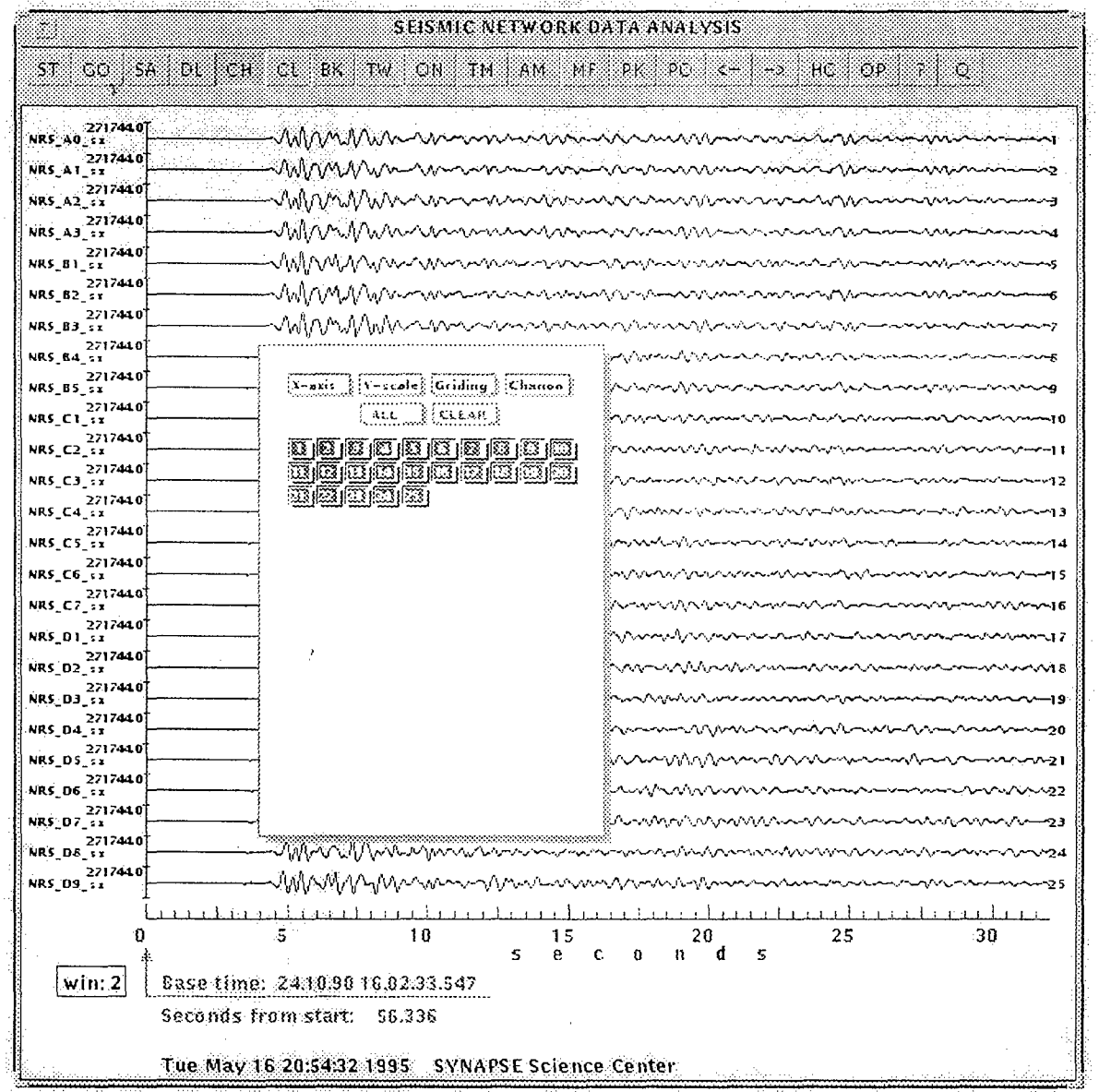


Fig.4. Selection of channels

dia

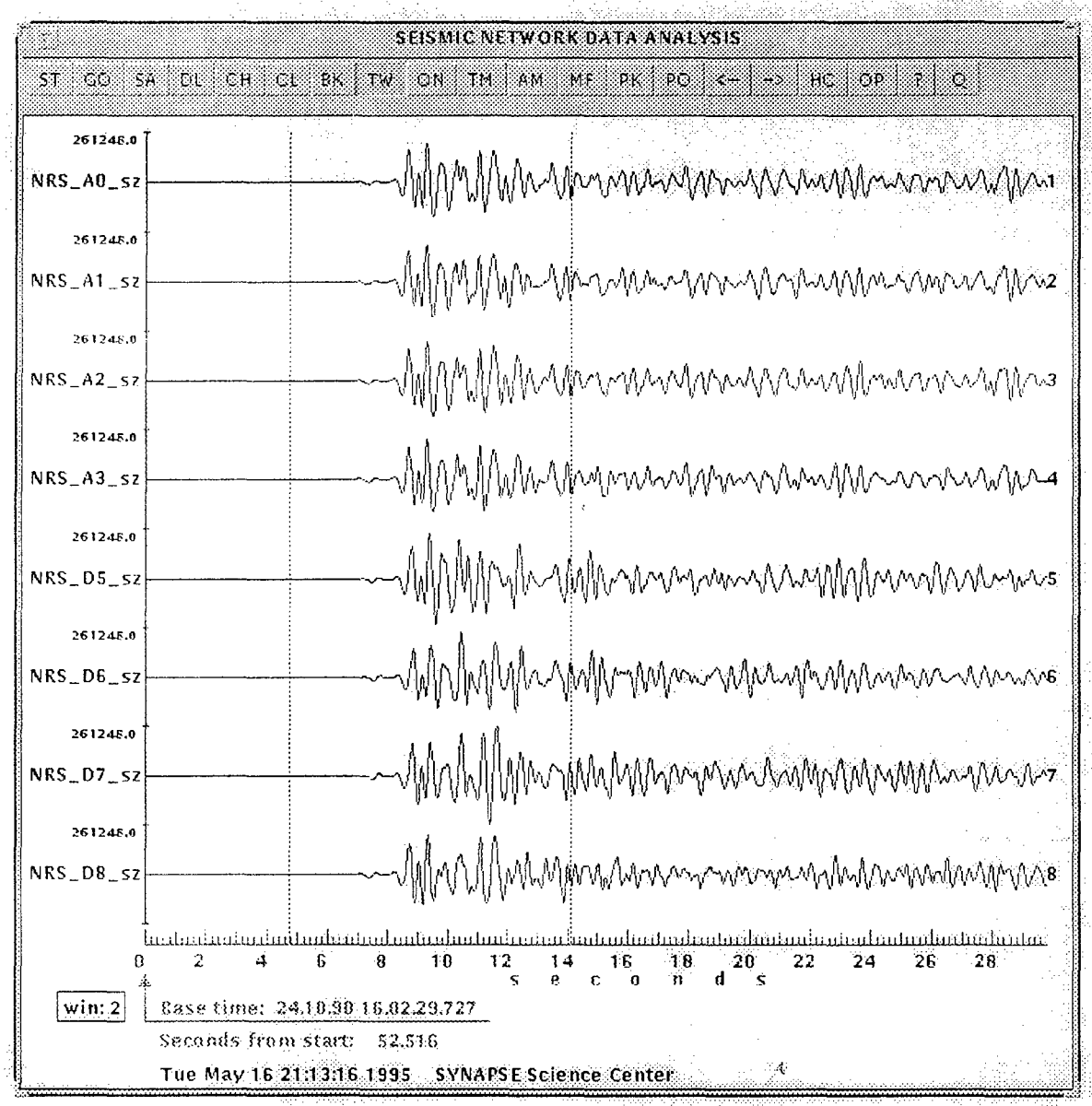


Fig.5. Set on window

dia

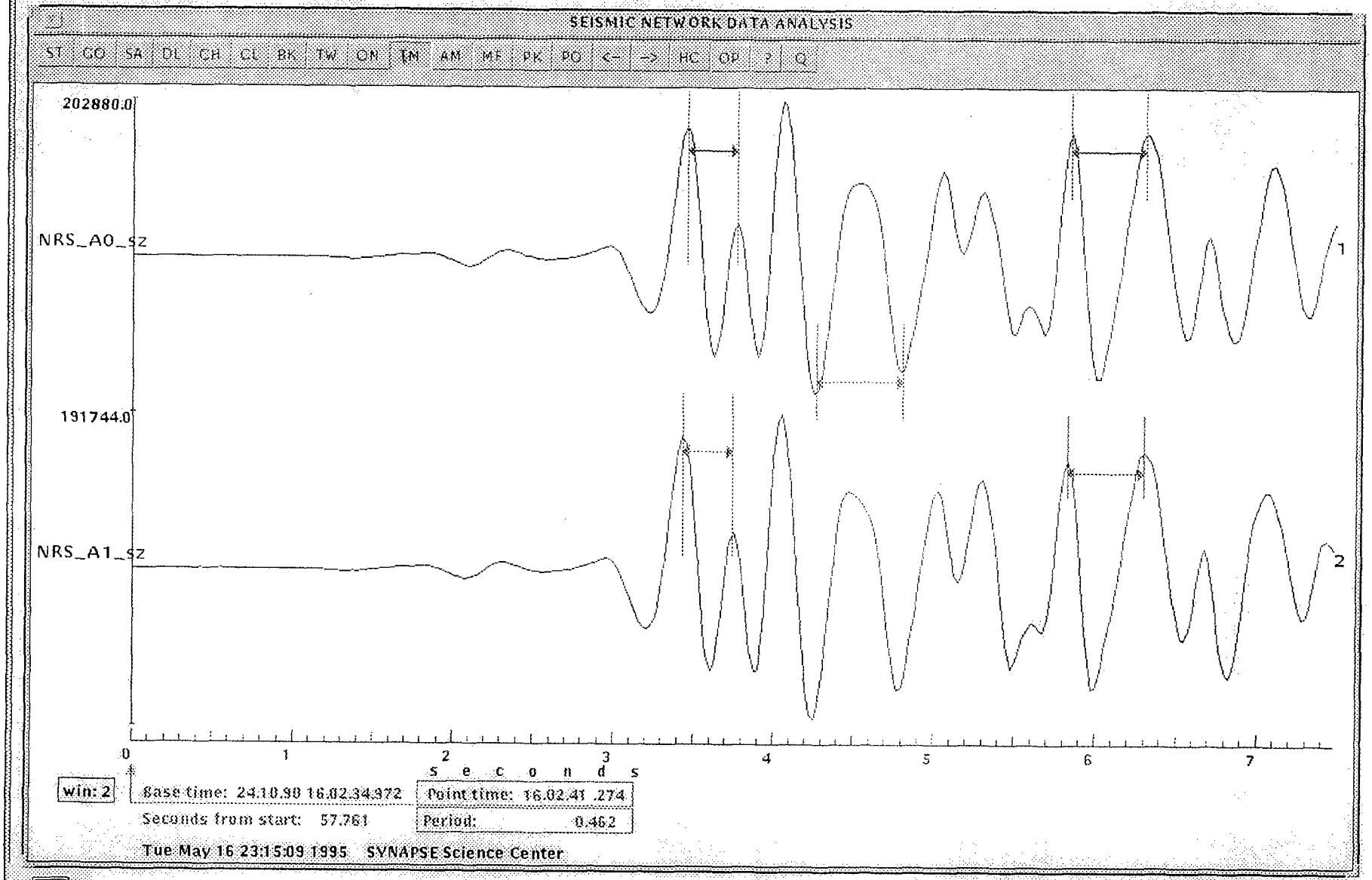


Fig. 6. Time measuring

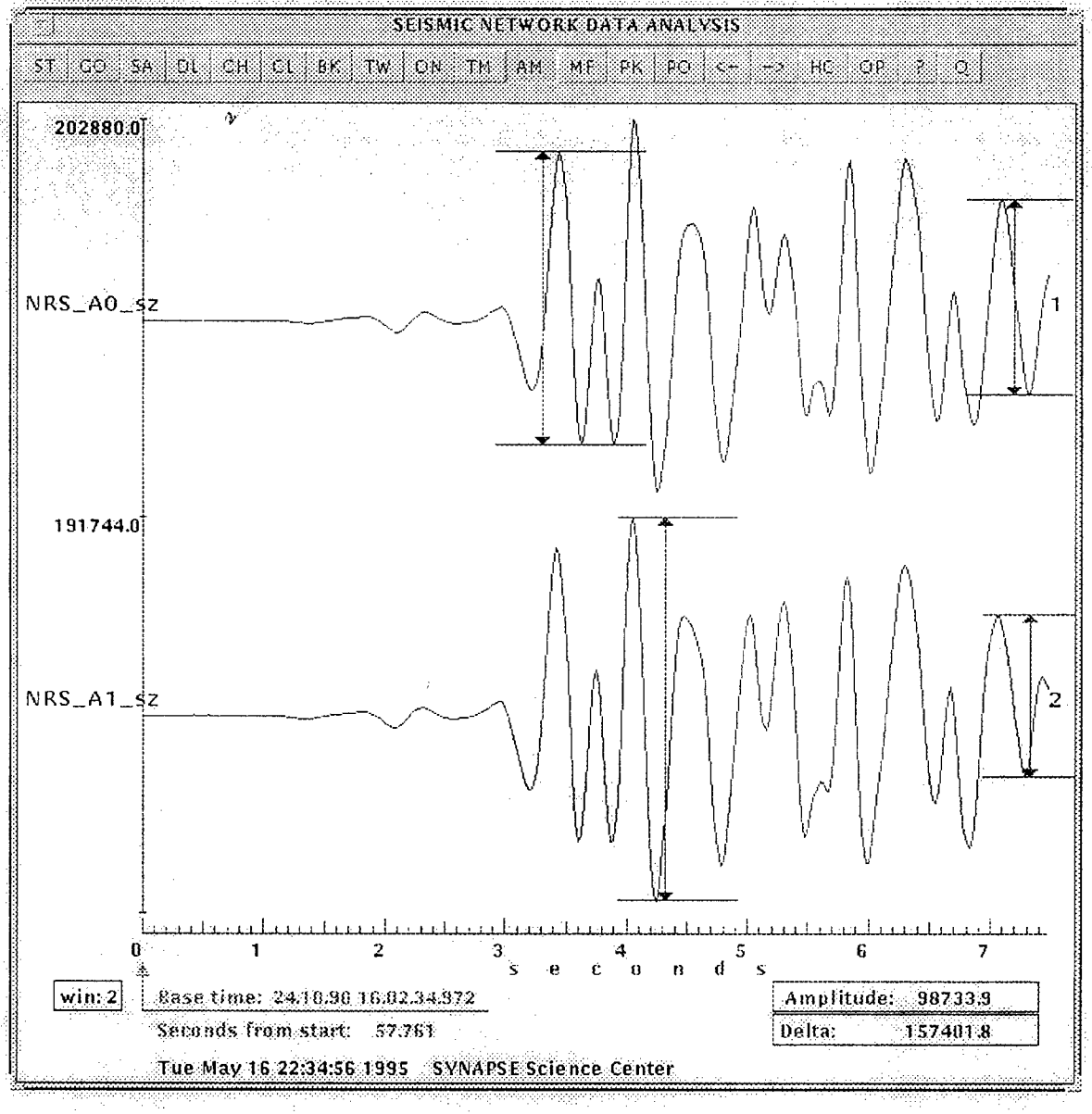


Fig. 7. Amplitude measuring

dia

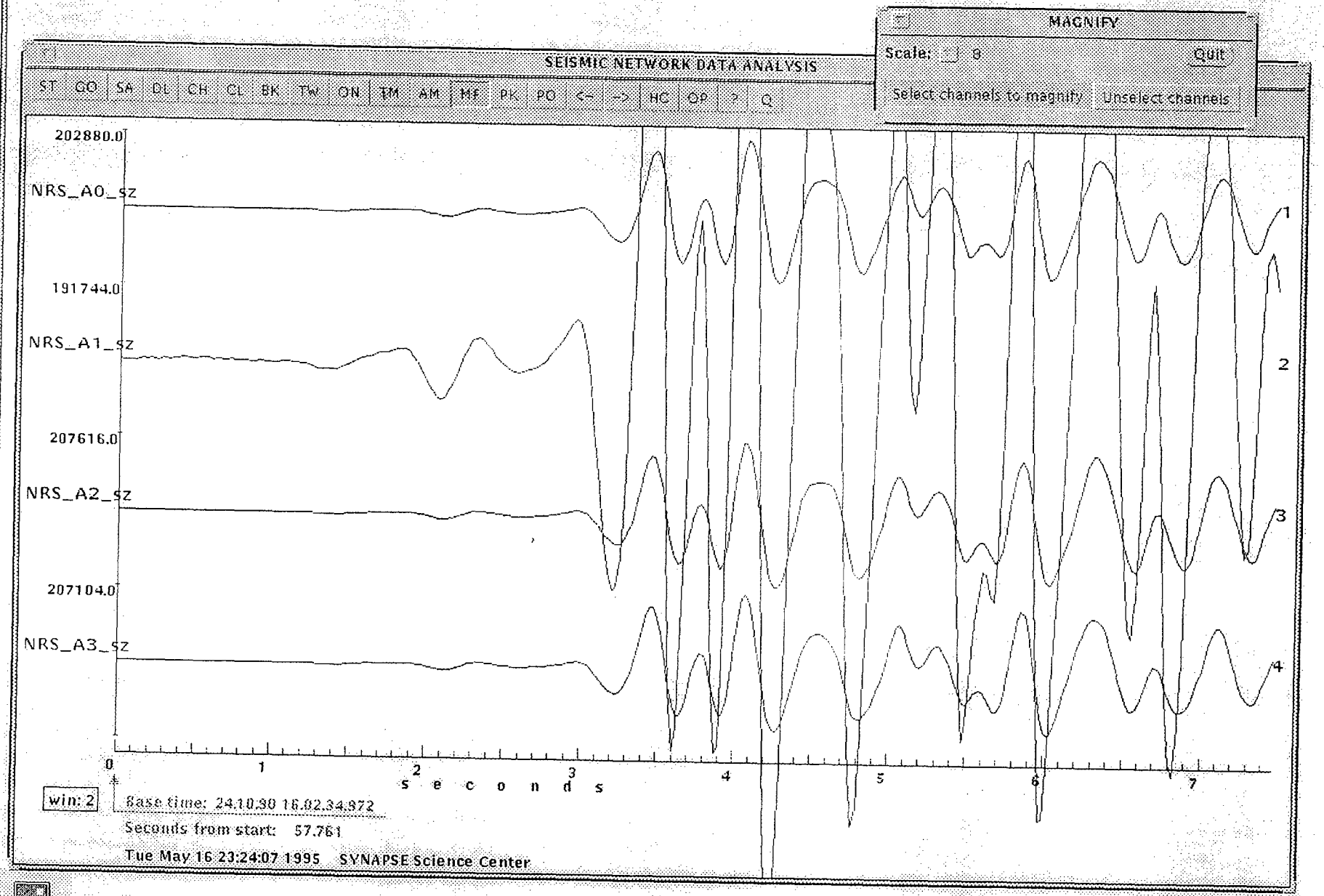


Fig. 8. Magnifying of 2-nd channel trace

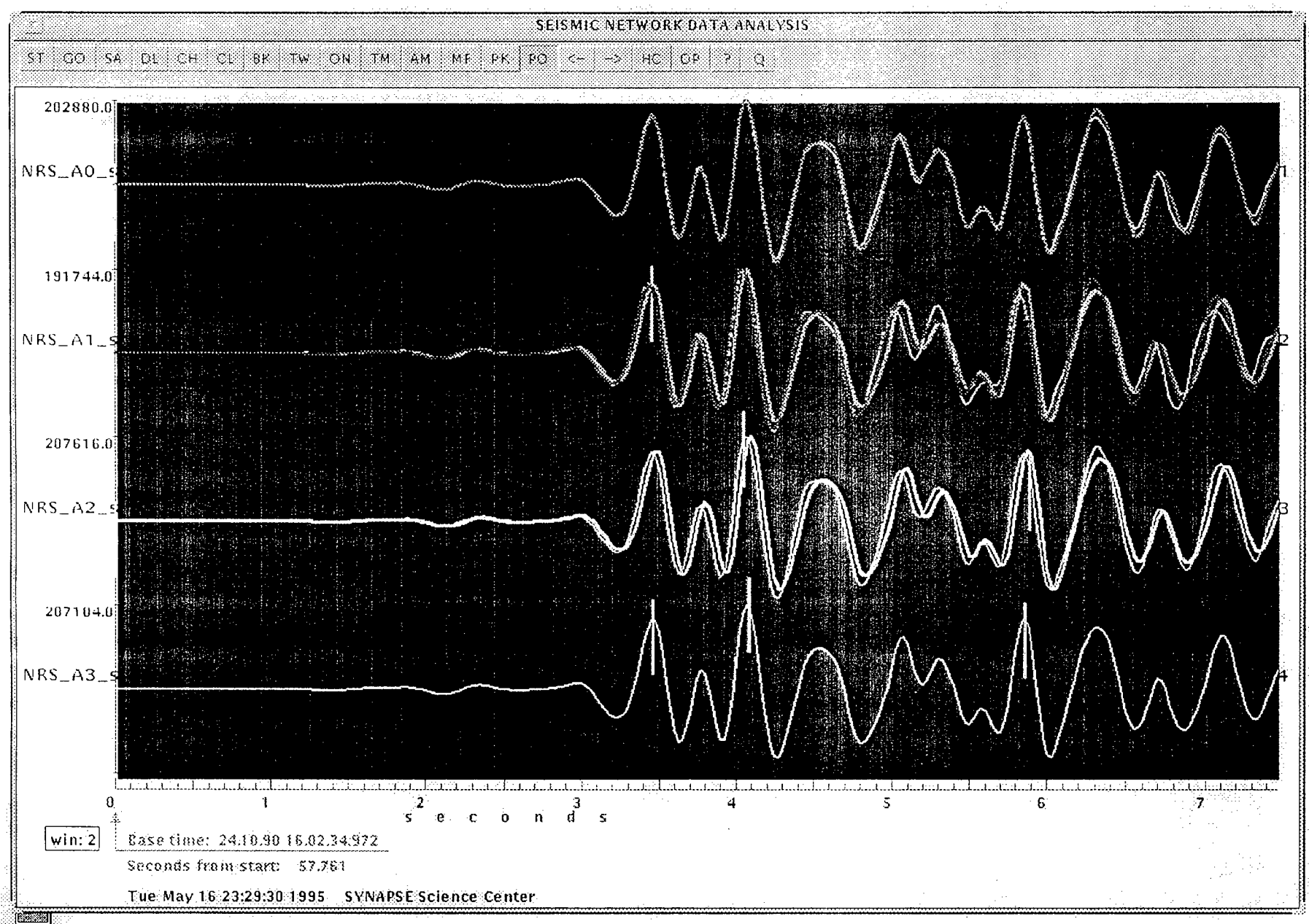


Fig. 9. Channel trace superposition

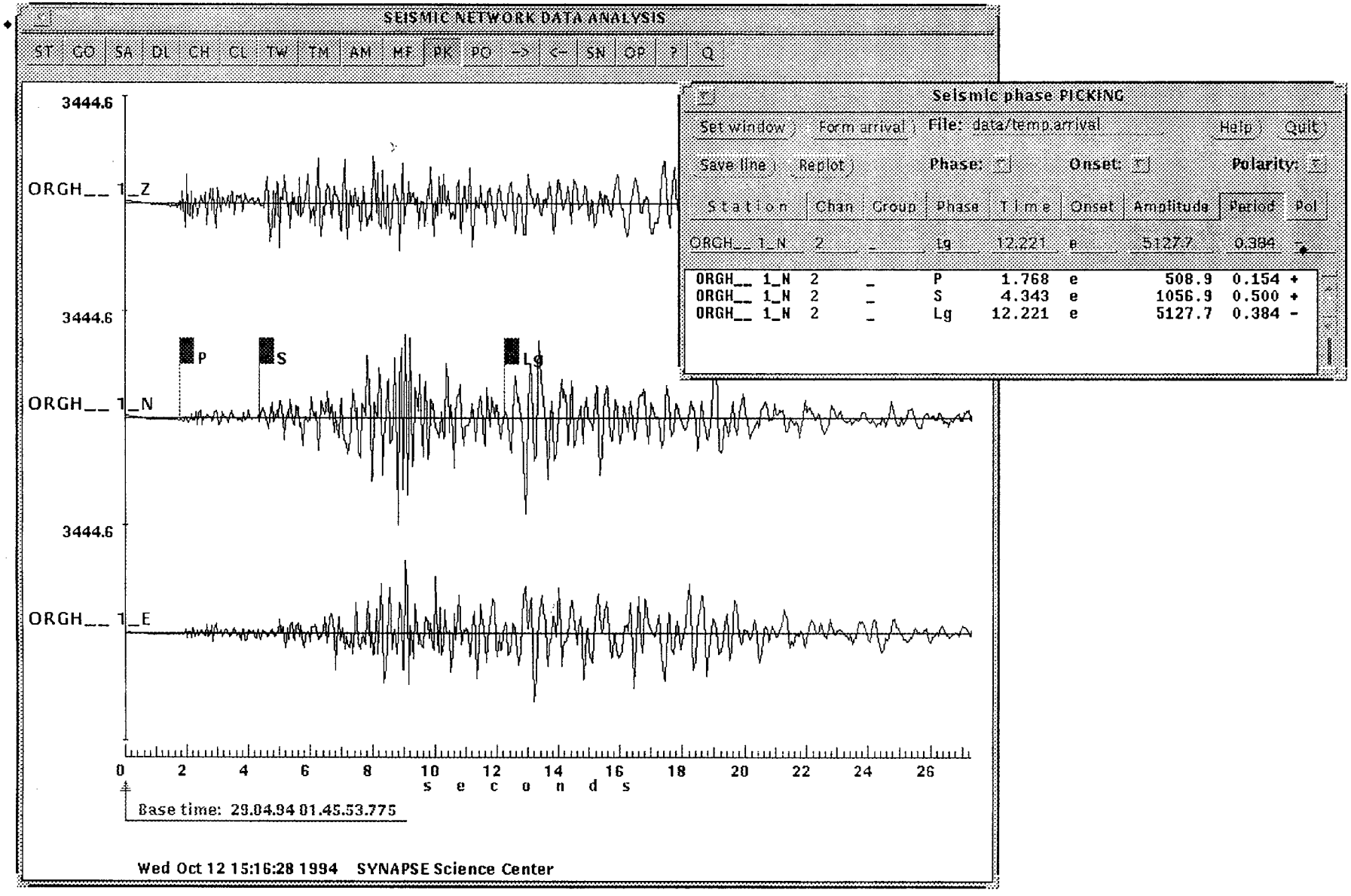


Fig. 10. Interactive seismic phase picking.

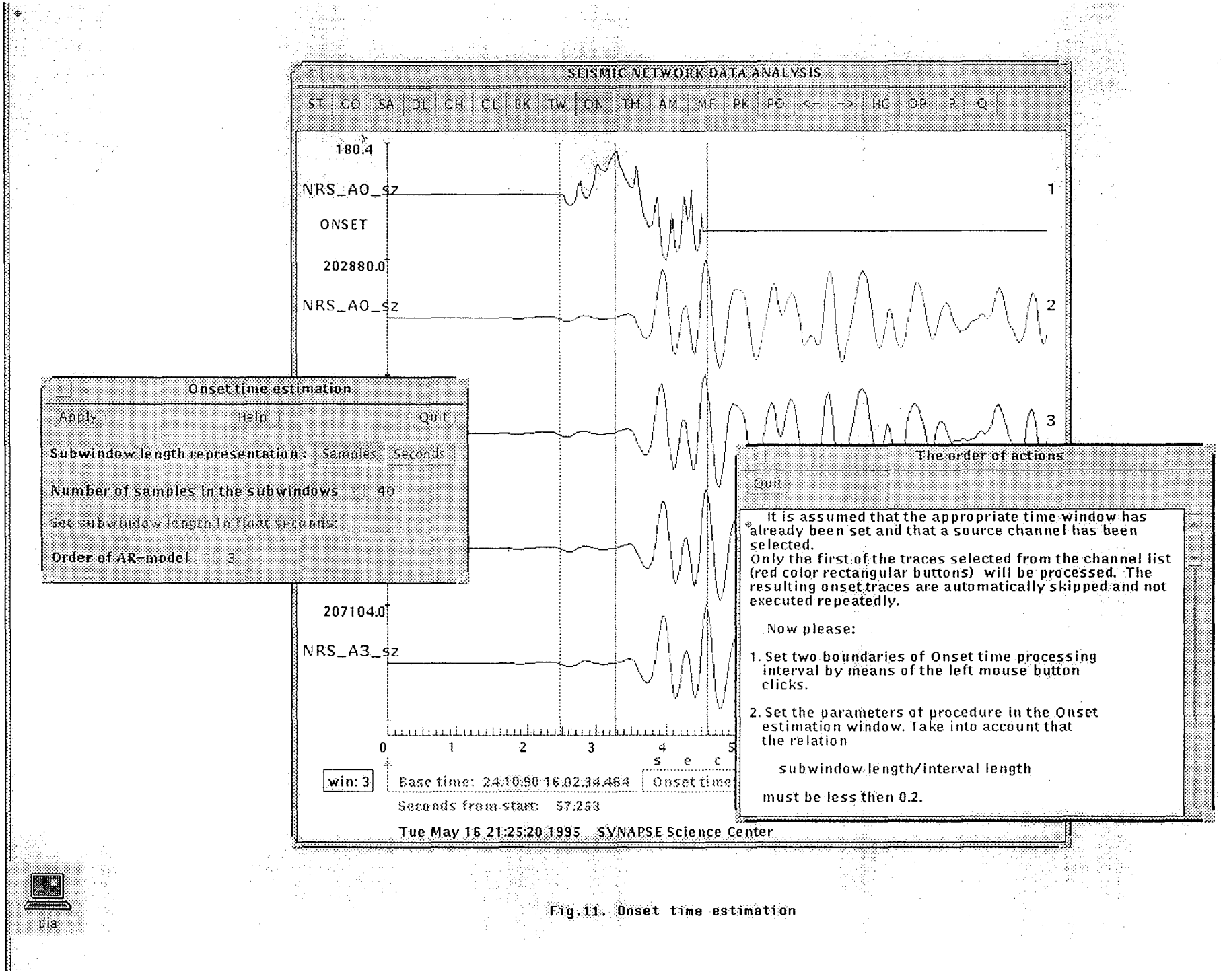


Fig.11. Onset time estimation

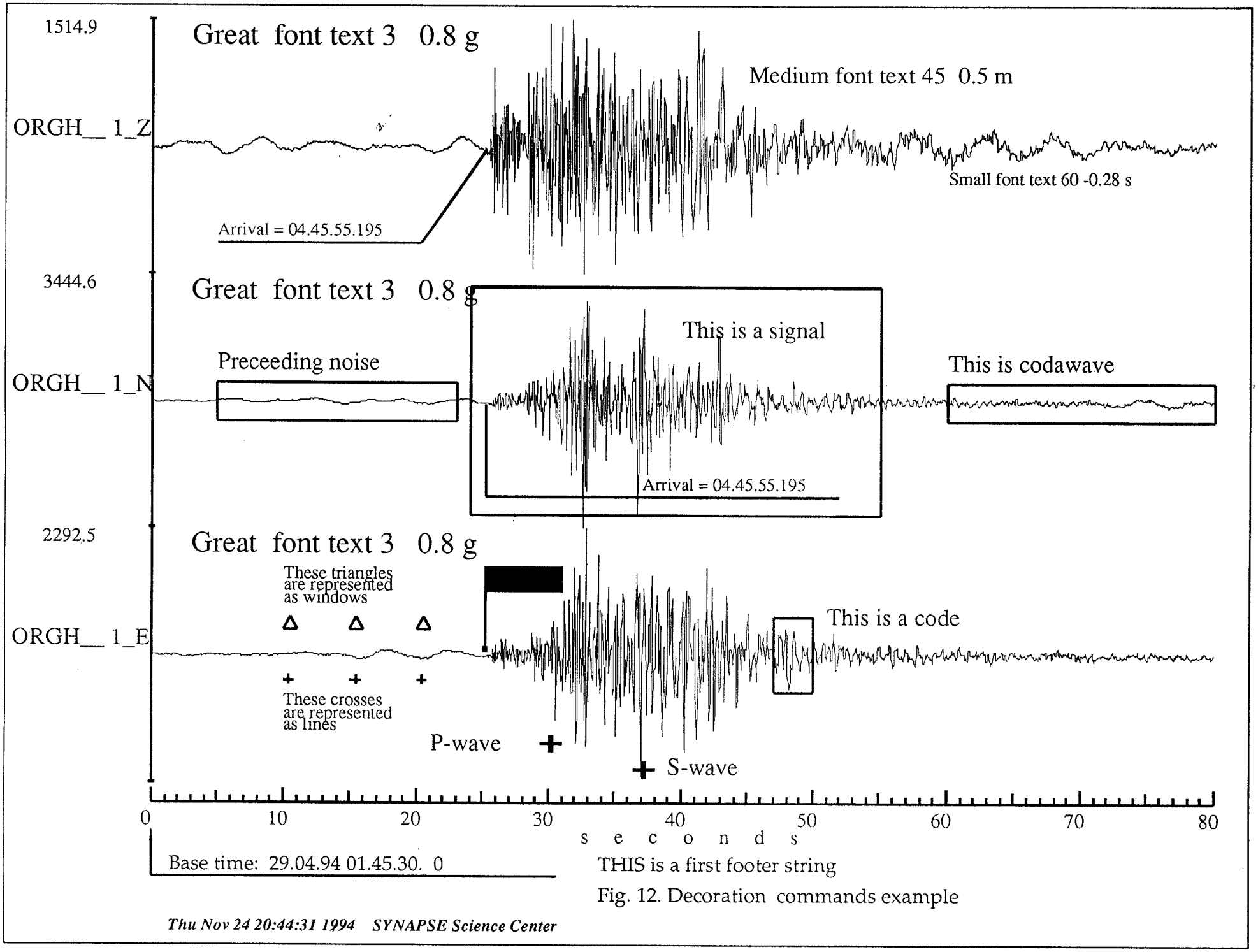
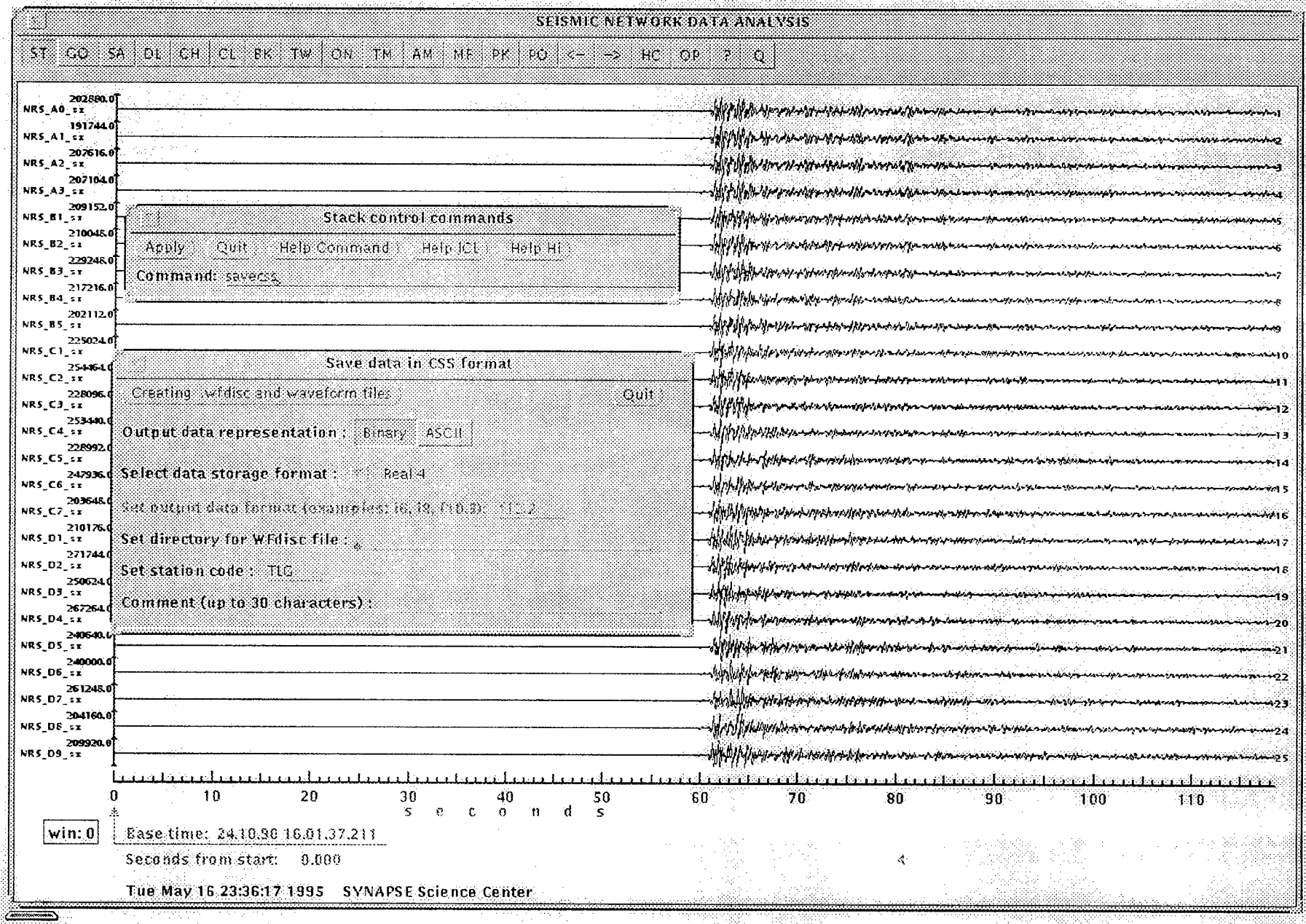


Fig. 12. Decoration commands example



dia

Fig. 13. Storing data in CSS 2.8 format

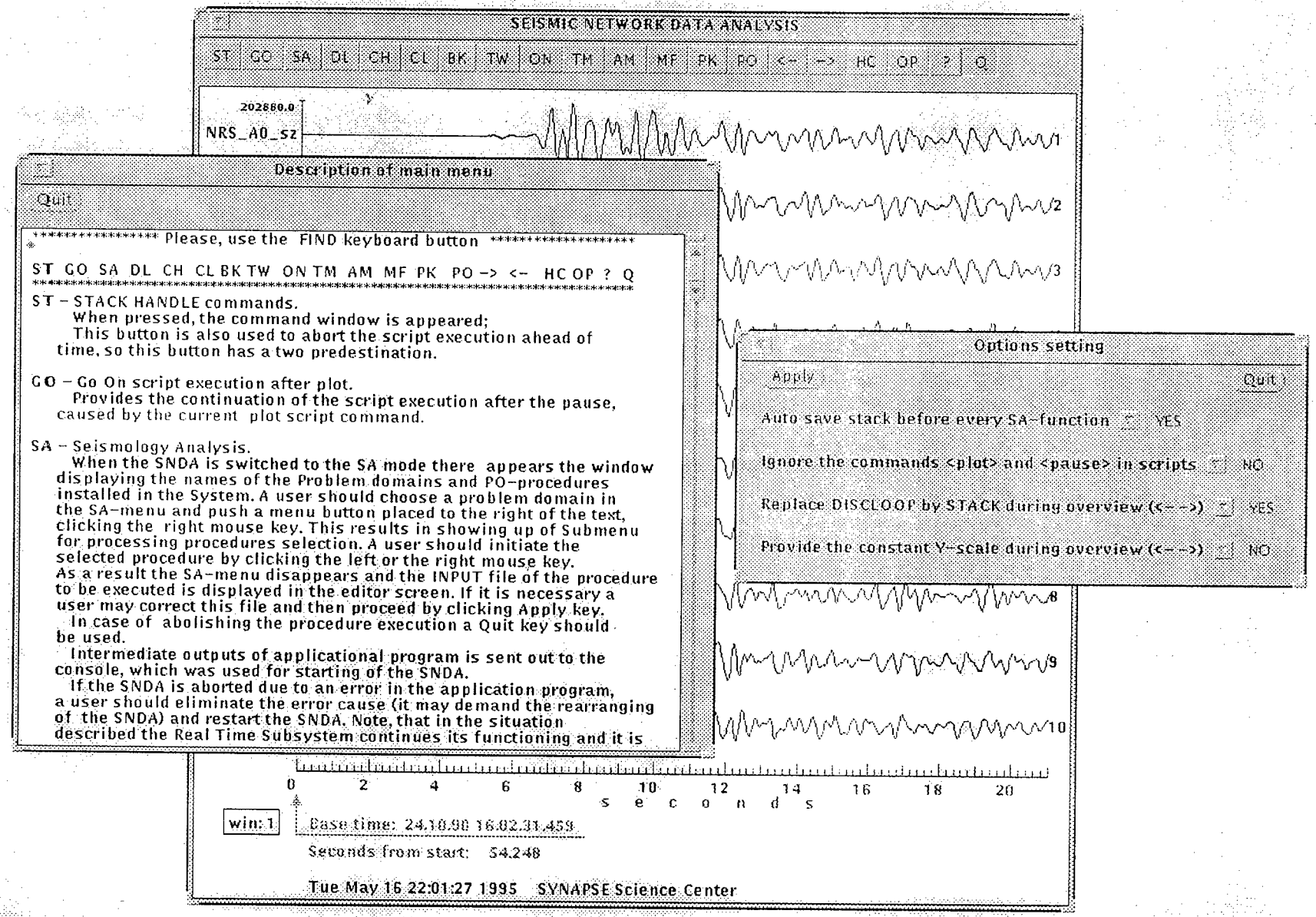


Fig.14. Service subwindows, ordered by click main menu "OP" and "?" buttons

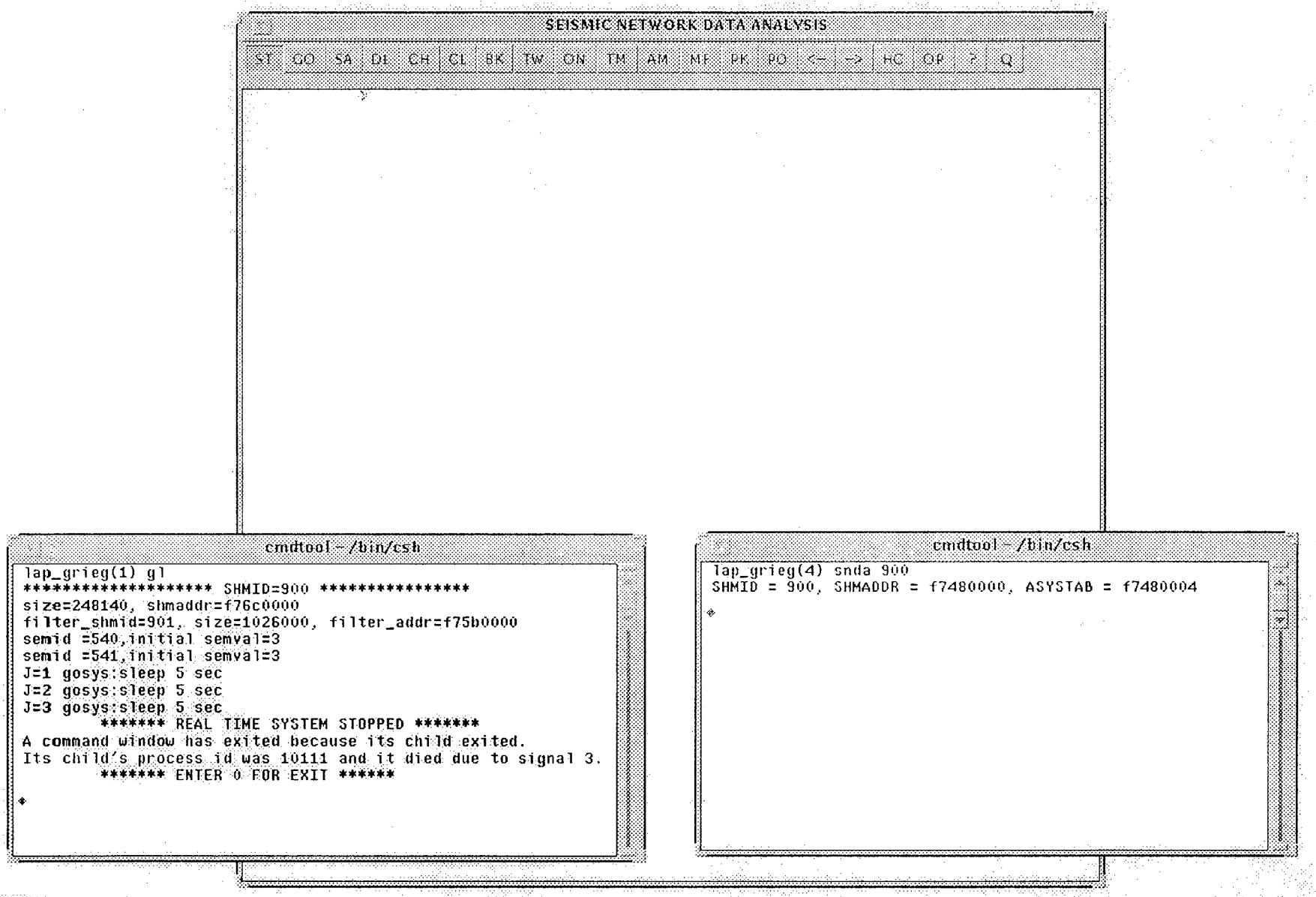


Fig. 15 System starting